UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Vlademir Fazio Santos

Improving the Performance of SJF/FIFO Hybrid-Scheduling Systems through the Management of Job Size

Melhoria do Desempenho em Sistemas de Escalonamento Híbrido SJF/FIFO através da Gestão do Tamanho de Jobs

Campinas

2019

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Vlademir Fazio Santos

Improving the Performance of SJF/FIFO Hybrid-Scheduling Systems through the Management of Job Size

Melhoria do Desempenho em Sistemas de Escalonamento Híbrido SJF/FIFO através da Gestão do Tamanho de Jobs

Thesis presented to the School of Electrical and Computer Engineering of the University of Campinas as part of the requirements to obtain the title of Doctor of Electrical Engineering in the area of Telecommunication and Telematics.

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica, na Área de Telecomunicação e Telemática.

Supervisor: Prof. Dr. Michel Daoud Yacoub

Co-supervisors: Prof. Dr. Edson Luiz Ursini and Prof. Dr. Paulo Sérgio Martins Pedro

Este exemplar corresponde à versão final da Tese de Doutorado defendida pelo aluno Vlademir Fazio Santos, e orientada pelo Prof. Dr. Michel Daoud Yacoub

---

Campinas

2019

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

Sa59i

Santos, Vlademir Fazio, 1959-
Improving the performance of SJF/FIFO hybrid-scheduling systems through the management of Jobs Size. / Vlademir Fazio Santos. – Campinas, SP : [s.n.], 2019.

Orientador: Michel Daoud Yacoub.
Coorientador: Edson Luiz Ursini.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Análise e desempenho de tarefas. 2. Sistemas aleatórios dinâmicos. 3. Escalonamento de processo. 4. Teoria das filas. 5. Processo decisório - Métodos de simulação. I. Yacoub, Michel Daoud, 1955-. II. Ursini, Edson Luiz, 1951-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Melhorando o desempenho de sistemas de escalonamento-Híbrido SJF/FIFO através da gestão do tamanho de jobs.
Palavras-chave em inglês:
Task Analysis and performance
Dynamic random systems
Hybrid scheduling
Mixed queuing networks
Decision making - Simulation methods
Área de concentração: Telecomunicações e Telemática
Titulação: Doutor em Engenharia Elétrica
Banca examinadora:
Michel Daoud Uacoub
Marli de Freitas Gomes Hernandez
Paulo Cardieri
David Bianchini
Roberto Colenci
Data de defesa: 28-02-2019
Programa de Pós-Graduação: Engenharia Elétrica

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0003-0994-9477
- Currículo Lattes do autor: http://lattes.cnpq.br/7813072475884435

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidato: Vlademir Fazio Santos RA: 801107

Data da Defesa: 28 de Fevereiro de 2019

Título da Tese: "Improving the Performance of SJF/FIFO Hybrid-Scheduling Systems through the Management of Job Size".

Título da Tese em Português: "Melhoria do Desempenho em Sistemas de Escalonamento Híbrido SJF/FIFO através da Gestão do Tamanho de Jobs".

Prof. Dr. Michel Daoud Yacoub, (FEEC, Campinas)
Profa. Dra. Marli de Freitas Gomes Hernandez, (FT, Limeira)
Prof. Dr. Paulo Cardieri, (FEEC, Campinas)
Prof. Dr. David Bianchini (PUCC, Campinas)
Prof. Dr Roberto Antonio Colenci (Centro Paula Souza, Botucatu)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação/Tese) e na Secretaria de Pós-Graduação da Faculdade de Engenharia Elétrica e de Computação.

I thank to The Being that endows the masters who I learn with and I pay tribute to them. I thank the gift of unmistakable and essential friendships (Fátima Gerra, Luiz Ariovaldo Junior e Osvaldo Torresan) and the opportunities and graces This Being wide opens in the way of my footsteps. I always learn little on my own and yet there is light in my path. God's privileges are beyond what one can comprehend. So the ones who say that God is above all, in fact, they are somehow confused because He is among all of us (Proverbs 15:3), and especially within all of us (Galatians 2:20). The infinite testifies Him, and once in a while it releases some quanta of dazzle. The beauty, the relief and the smile always arrive. Not always together!

Thanks, once more ...

Acknowledgments

"This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001"

I would like to thank my advisor Professor Michel Daoud Yacoub (Ph.D) for making this project feasible, for his patience and respect for my frailties. I deeply thank my co-advisor Professor Dr. Edson Luiz Ursini for his time, professionalism and his continued generosity. My sincere thanks to Professor Paulo Sergio Martins Pedro (Ph.D) for his contribution.

"Archimedes is correct. But even if a lever and a point of support can move the world, what would all this be without the spiritual moment?"

Abstract

This thesis proposes a discrete-event simulation method to plan and manage the performance of M/G/1 queuing systems of relative complexity. The system has parallel servers undergoing dynamic changes under system instabilities (e.g., spontaneous oscillations of the incoming traffic to the system overload). Entities have multi classes of jobs and each server performs a single class of this jobs. The method manages the size of jobs that may cause loss of performance, e.g. the delays in average residence time. Performance management is carried out via the monitoring of the impact of classes of job sizes on the total system delay. Jobs that exceed a certain threshold value may then be managed accordingly, e.g. by moving them to different servers or by (temporarily or permanently) blocking them. We present a case study of loading and moving cargoes within an industrial production area. Each cargo consists of multiple product classes which are simultaneously loaded on different servers. This logistic-model implementation begins with a well-known validated model extended by small validated increments to better be able to represent the real-world. The technique of Improving the Performance of SJF/FIFO Hybrid-Scheduling Systems through the Management of Job Size under dynamic conditions, i.e. when subject to toggling SJF and FIFO policies and fluctuations of inbound traffic, has shown to be effective in the reduction of the time average, in the decrease of the mean time maximum, and in the identification of jobs that cause delays to the system, and thus enable the management of these jobs to mitigate unwanted system performance.

Keywords: performance analysis; dynamic models; performance management; stochastic control systems; hybrid scheduling; mixed queuing networks

Resumo

Esta tese propõe um método de simulação discreta para planejar e gerenciar o desempenho de sistemas de filas M/G/1 de relativa complexidade. O sistema é formado por servidores em paralelo submetidos a mudanças dinâmicas entre as políticas de escalonamento SJF e FIFO. O tráfego de entrada de entidades é aleatório com oscilações até a sobrecarga do sistema. As entidades são formadas por multiplas classes de jobs e cada servidor processa uma única classe desses jobs. O método gerencia o desempenho das classes de jobs que provocam perda no desempenho do sistema por atrasos no tempo médio de residência. O modelo de simulação obtém o tempo médio de residência relativo de cada classe de job para calcular o atraso relativo dessas classes no atraso total do sistema. Os jobs que ultrapassam os limites dos requisitos podem ser gerenciados, e.g., direcionados para outros servidores, ou serem bloqueados temporária ou definitivamente. Como exemplo de um problema complexo, apresentamos um estudo de caso logístico de carregamento e movimentação de cargas dentro da área de produção industrial. As cargas são formadas por múltiplas classes de produtos simultaneamente carregados em diferentes servidores. A implementação desse modelo logístico é iniciada com um modelo reconhecido e validado e prossegue com pequenos incrementos validados até a representação de um modelo o mais próximo possível da realidade. A técnica de Escalonamento Híbrido de Sistemas com Gestão do Tamanho de Jobs permite a mudança dinâmica de políticas de escalonamento do sistema entre SJF e FIFO, ainda que sujeita a variações abruptas de tráfego de entrada. Essa técnica é efetiva para reduzir os tempos médios, conter os tempos máximos e habilitar a identificação dos jobs que provocam atrasos, permitindo dessa forma, ações de gestão para mitigar resultados indesejados.

Palavras-chaves: análise de desempenho; modelos dinâmicos; gestão do desempenho; sistemas de controle estocástico; escalonamento híbrido; redes de filas mistas.

List of Figures

List of Tables

List of Acronyms, Abbreviations and Previous Definitions

$\frac{1}{\lambda}$ interarrival time

λ arrival rate

$E[B_i] = \frac{1}{\mu_i}$ mean service time for server i

$E[R] = \sum E[R_i]$ total mean value for residence time for all servers

$E[R_i]$ mean value for residence time due to server i

TEX replication time divided by # *of outbound units*

TEX_{limit} replication time divided by # *of created units*

3D three dimension

3PL third party logistics

CSM continuous simulation

DES discrete-event simulation

e.g. exempli gratia

HW Half-Width is the distance from the mean to the edge of the confidence interval

i.e. id est

InV incremental validation

JQN Jackson queuing networks

JSQ joint-shortest-queuing policy

M&A Merge & Acquisition is the process of consolidation of companies

MQN mixed queuing networks

mt metric ton

MTO make-to-order

MTS make-to-stock

NASS National Agricultural Statistics Service

QoS Quality of Service

R$ brazilian currency (Real)

RFID radio frequency identification

SJF shortest-job-first

SP Brazilian São Paulo State

SRDQ SJF-RR-Dynamic-Quantum Algorithm

TAVG simulated average residence time

TMAX simulated average maximum residence time

TSTD simulated average standard deviation for residence time

US United States of America

USDA US Department of Agriculture

ut units of time or time units

VIP Very Important Person

Summary

1 Introduction

For decades, the impact of logistic processes on value chains has been discussed, both in the positive and in the negative perspectives. The most important positive perspective is present in Yee e Oh (2013) in which, between 1980 and 2010, there is a global appreciation in reducing costs and adding technological value by transferring those non-core enterprise activities to other companies. This practice is known for business strategies as, for example, outsourcing and offshoring, among other strategies generally associated to the globalization process. The outsourcing is understood as a management strategy in which companies/institutions hire activities (from other companies) that are not in its core business. An example of outsourcing is practiced by the automotive industry that hire logistics enterprises (the third-party logistics enterprises (3PL)) to transport their products. The offshoring strategy means the practice of basing some of the organization's activities overseas, or transferring this activities (including the labor force) to other countries, so as to take advantage of lower costs (BLINDER, 2009). One well-known example of offshoring during the last decades (2000-2010) is the initiative of Ford Motors Co. in transferring some of its activities to Mexico, India and other countries. The negative perspectives include examples of technological capital flight, infrastructure and knowledge deficiencies, excessive dependence on a transportation modal, lack of adequate public policies, etc. These negative perspectives are pointed out as important causes of recurrent problems in countries with economic and social characteristics similar to Brazil. Examples are presented in the case of misguided public policies that are eventually diverted to favor oligopolies (ARAÚJOJR., 2015). Another example presented by (CORREA; RAMOS, 2010) are the bad conditions of conservation of transport infrastructure and the distribution of soybean from producing regions that add to the concentration and dependence of the road modal. The Brazilian crisis of cargo transport, with the strike/lockout started on May 21, 2018, made clear this set of deficiencies, and its consequences are no longer alerts from experts and became a historical case of failure of the public structure (OGLOBO, 2018a; OGLOBO, 2018b; ENPNEWSWIRE, 2018a; ENPNEWSWIRE, 2018b; DARLINGTON; ANDREONI, 2018).

1.1 Insourcing as an improvement strategy

The failure of the public structure and the effective tendency to reduce logistics costs can be observed in the contraposition of the dependence on third parties. Although rarely observed in countries as Brazil (as evidenced below), it may be similar to other world's events,

such as a worldwide trend of the last decade to value the insourcing (WARNER; HEFETZ, 2012; CHUDZICKA, 2013; REKHA, 2017; NOAM; GELBARD, 2018). Insourcing contrasts with Outsourcing as a management strategy in which companies/institutions oppose to hire activities from other companies, or in other countries (Offshoring). Insourcing is the movement of re-absorption of activities into the boundaries of a company, or in the country of the company's origin (OXFORDDICTIONARIES.COM, 2018; BUSINESSDICTIONARY.COM, 2018).

This phenomena in Brazil happens as opposition of the outsourced services and/or activities that imply institutional dependence or impairment of the core activity's quality (FERNANDES *et al.*, 2016; MARTINS, 2016). There is a lack of this subject's studies and (REIS *et al.*, 2018) made a systematic review where they reported one case of the Brazilian mining area and another case in the accounting services sector. Another case reported by the Portuguese group Sovena (www.sovenagroup.com/pt/), its activities in Brazil during the year of 2016 adopted a well-succeeded insourcing model that eliminated operational difficulties and consolidated its quality model in its food segment (LEADERSHIP, 2016). Santos e Teixeira (2015) studied the success of implementing insourcing in the industrial maintenance service. This implied increasing internal capacity for controlling over costs, quality and knowledge. The Brazilian crisis of truckers (SANTOS *et al.*, 2018) also gave importance to the logistic component of this issue.

1.2 The impact of Loading/Unloading/Queue process

Pera *et al.* (2018) exemplified the impact of costs on logistics processes. They showed the most important cost components (in R$/t.km − Brazilian Real currency per ton per kilometer) of soybean's road-transportation in the distance of 100km away with B-train trucks (vehicles with two trailers linked together). We chose some of these costs in Table 1:

Tabela 1 – Some important components of the logistics cost of soybean's road-transportation for distances of 100km with B-trains (37t). Data based in Pera *et al.* (2018).

Type	Cost in R$/t.km ... Category	
	minimum	maximum
fuel consumption	0.17 ... 3.00 km/l	0.19 ... 2.00 km/l
highways' quality	0.17 ... 70 km/h	0.20 ... 30 km/h
capacity of vehicles		0.16 ... 37 ton B-train
loading | unloading | queue	0.12 ... 2h process	0.26 ... 10h process

These data show that the cost of time wastage to loading/unloading and waiting

queues for short transportation can be from 33 % to 37 % higher than the individual values of each of the most significant transport cost components analyzed by Pera *et al.* (2018). To assess the magnitude of the waste in the logistics process, we highlight recent data from the US Department of Agriculture (USDA). The USDA (2017) and USDA (2018) reports compare the performance of soy exports between the US and Brazil in the 2016-18 period. They point out that the transport cost practiced in Brazil during 2017 were 118 % higher than those practiced by American ports when compared with similar distances from the production areas. By comparing data from the Port of Santos from Sao Paulo, Brazil, with its corresponding American competitors, we find that the average cost for the port of Santos is R$ 30.87 /mt (Brazilian currency per millions of tons) against R$ 14.16 /mt for the Americans. Moreover, the prices practiced by American producers for the European market surpassed by more than 20 % those that are practiced by the Brazilian producers: what makes the American producer more profitable, with higher productivity. When we consider the cost of the wastage of time relative to loading/unloading and waiting queues pointed out by Pera *et al.* (2018), one can reach an expressive reduction of the commodities' prices with increased financial results per ton for the Brazilian producers.

1.3 Motivation

This lack of structures, lack of specific technologies, lack of applied knowledge which are notoriously perceived and measured, motivated us to present a solution to reduce the time wastage related to loading/unloading and queuing process (Fig. 1).

This initiative seeks better results for the performance of the logistics process. This thesis contribution basically addresses queue systems with several servers in parallel, each one with an independent operation. Figure 1 shows a sequence of images of an actual example of this process. The location is an industrial plant designed as vertical buildings (Fig. 1(1)B) in a great urban port areas (Fig. 1(1)A), surrounded by viaducts, railroad and highways (Fig. 1(1)C) and consequently, with outside space restrictions (Fig. 1(1)E) and inside space constraints too (Fig. 1(1)D). Figure 1(2) shows the entrance of a truck in an inside-loading area of this typical industry plant. Figure 1(3) shows the vehicle crossing down one of the processing buildings (e.g. the upper processing floors of Fig. 1(1)B) and in Figure 1(4), the truck is in a short space maneuver to pulling up for loading its cargo as shown in the next images. Figure 1(5) allows us to realize a loading process in a discharge hopper facility of solid grains. Similarly, Figs. 1(6), (7) and (8) show a loading process of bagged grain commodities in a human-operation discharge (Figs. 1(6)B, (7)B and (8)) facility supported by conveyor belts (Figs. 1(6)A and (7)A). This kind of human-charge/discharge operation in urban centers is

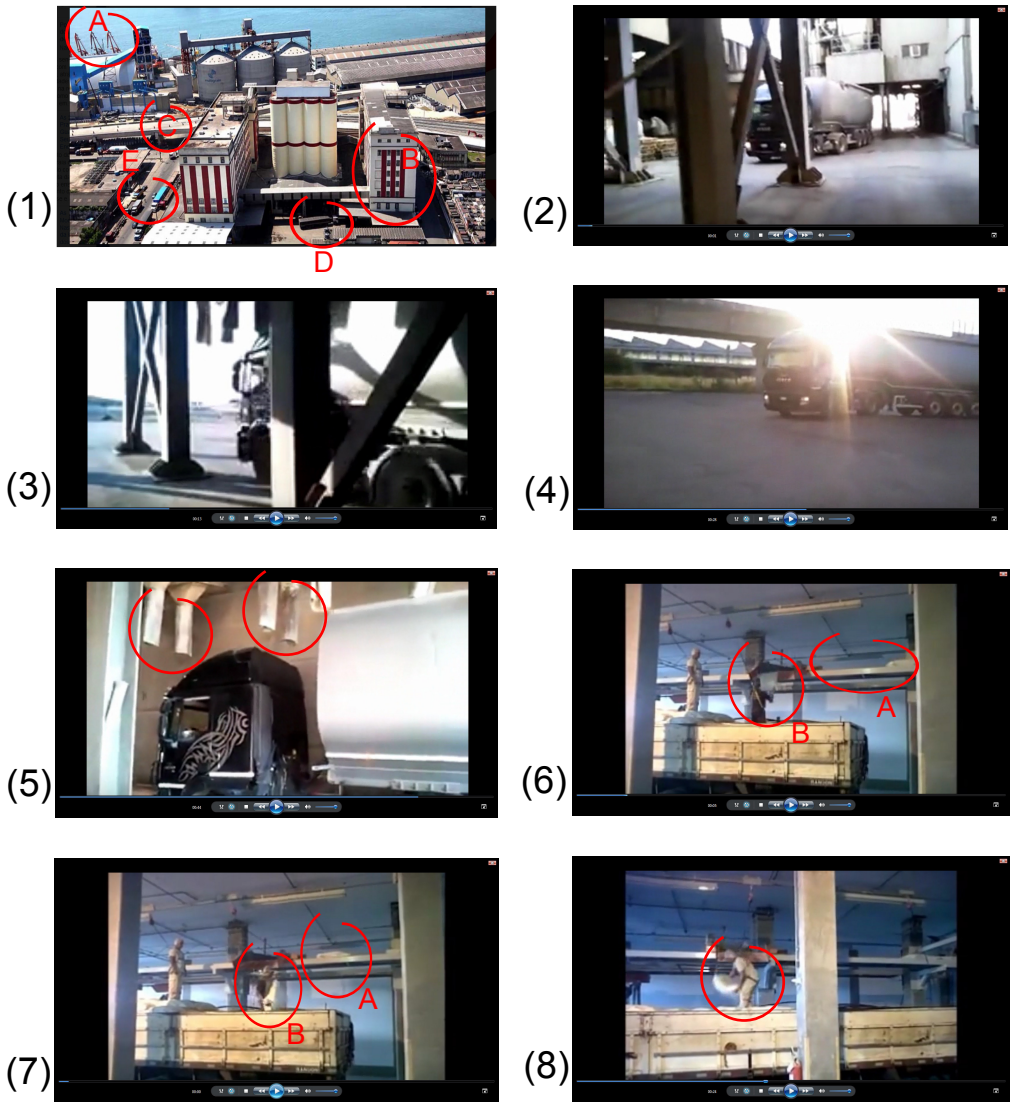


Figura 1 – Sequence of 8 images as an example of Loading Processes inside industry plants of urban areas.

usually handled by the retail market, in order to process a set of consummation commodities, e.g. food grains or its derivatives.

This work focuses on this kind of human-charge system in which the sizes of jobs (e.g. the amount of a product from a set of products in a cargo) are known (or can roughly be estimated) and non-preemptive queuing disciplines are employed. We began the investigation by the concept of predictability brought by Wierman e Harchol (2005) and Mor (2010) that can be understood as the system capacity of consistently identifying different job's sizes despite of changes and variations. It is directly associated with fairness and it may or may not cause a positive impact on the system throughput and mean average times.

Another aspect of the complex system to be evaluated is the fact that the entity (here

said as a cargo) to be completed must have jobs performed on all servers and each server performs one different and only kind of job. An example of this type of problem is the case of a truck that must be loaded with three different types of commodities, for example maize, beans and soybeans. Loads are executed through three different servers, independently until the task (truck loading) is completed. The goal of the thesis is to manage the size of jobs in a complex system to increase the flow of system entities while keeping (or increasing) the quality of service (QoS) established by certain requirement(s).

This thesis is composed of eight chapters. The main concepts upon which the thesis is based are: discrete-event simulation (DES), Jackson queuing networks (JQN), incremental validation (InV), hybrid policy, dynamic toogling, and the analysis and management of the performance of jobs' sizes. Chapter 2 is the article entitled *"A Management Tool Based on Discrete Event Simulation for Humanitarian Support"* published at the Winter Simulation Conference 2018 (Gothenburg, Sweden). It represents the initial discrete event simulation model which was validated by an analytical model based on Jackson Queuing Networks (JQN). The model consists of a system with parallel servers, job reentries, and closed-queuing networks in its core. The main application is an humanitarian service where the management aspect is essential for its performance.

Chapter 3 entitled *Identifying the SJF and FIFO Compromise* implemented the incremental evolution (InV), whereby one model is incremented (e.g. one added functionality) from a previous validated model. The first part of the chapter evaluates the system subject to the non-switching, full policies (either 100% SJF or 100% FIFO). The second part of the chapter also evaluates the trade-offs of the variables subjected to SJF and FIFO.

The article in the Chapter 4 entitled *"A New Framework for the Performance Analysis of the Single Server Non-Preemptive Scheduling under Varying Traffic Conditions"*, was submitted to The Scheduling Journal (JOSH, Springer/Nature). It describes how the SJF policy contributes to the effective analysis of the performance of systems submitted to instabilities. This contribution is studied through a data structure shared with both discrete simulation (DES) and continuous simulation (CSM) for a queuing system of one single server. Chapter 5 entitled *"Simulation Model for Performance Analysis of a Hybrid-Policy (SJF/FIFO) Parallel-System"* looks into the DES model toggling the SJF and FIFO policies for better system performance of multi-class entities in multiple parallel-servers. The concept is based on JQN, InV and SJF. The paper shows the DES model, its implementation, and the results of the dynamic toggling of the operation mode of a system, named *Hybrid Selective SJF/FIFO scheduling*. This policy is *hybrid*, because in one single inflow/outflow cycle (Section 5.3.2), it is possible to execute two scheduling policies, i.e. SJF and FIFO. It is also *selective* because it defines which policy, at what point in the cycle and, at what intensity it is applied to the

system. We define the *intensity* of the selected policy as the maximum number of jobs allowed in the system during the selected policy period. The results allow us the identification of two key perspectives for the model: planning/sizing and management. From the planning perspective (Section 5.3.1), it is possible to understand the behavior of the output variables under a complex scenario, e.g. with varying traffic conditions and hybrid scheduling (SJF and FIFO). The model also identifies the compromise intervals - positive, negative and null - between the variables. From the management perspective, the contribution is the elaboration of an operational management map that allows the manager/analyst to anticipate possible bottlenecks and, thus, the possible strategies to be implemented. Therefore, it allows the improvement of the system operation according to the set requirements. Chapter 6 entitled *"Identifying Jobs' Sizes Performance in the Hybrid SJF/FIFO System"* shows a model for the classification of jobs' sizes through the jobs' delay in two levels to be performed with DES. It uses the same simulation model of the Chapter 5. However, it expands the idea of classification of mean times to handle the classes of mean-times with the largest identified delays. The jobs that fall within the latter mean-times are also classified. By having the classification and identification of jobs that most negatively impact the performance, it is possible to manage them accordingly in order to improve system performance.

The Chapter 7 summarizes the key results and introduces a discussion on the preceding chapters and their link to the main results of the thesis. Finally, in the last chapter (Chapter 8) the concluding remarks are presented with suggestions for future work.

2 A Management Tool Based on Discrete-event Simulation for Humanitarian support

Full paper presented at: The Winter Simulation Conference 2018 (WSC'18) INFORMS/IEEE December 9-12 Gothenburg, Sweden

Humanitarian aid is material or logistical assistance provided for humanitarian purposes, typically in response to humanitarian crises including natural disasters and man-made disaster. Humanitarian assistance requiring short response time windows in almost the whole world may be subject to long queues due to managing problems, e.g., the lack of control and/or inefficient infrastructure. This work tackles such challenge by proposing a low-cost planning and managing model and method based on a discrete-event simulation mirror connected through WEB tools to a near or far management level. The usual configuration of parallel servers (for instance, supported by local $RFID$ monitoring) is implemented by a discrete-event simulation model that is validated by Jackson Networks (and vice versa). The results show a flexible model that may identify bottlenecks in advance in order to accommodate traffic flow variations.

Keywords: managing parallel queues; Jackson Queuing Networks; RFID monitoring; traffic variations

2.1 Introduction

Humanitarian aid consists of material and logistic assistance to people who need help. It is usually short-term help until the long-term help by government and other institutions replaces it. The primary purpose of humanitarian aid is to save lives, reduce suffering and respect to human dignity. In one particular case in Rio de Janeiro, Brazil (OGLOBO, 2018a), a crowd of more than 13,000 people were given assistance of humanitarian nature during less than half a day.

The five most relevant issues that Richardson *et al.* (2016) concluded in their research on inventory prepositioning for humanitarian logistics are (in order of importance): the speed of response, physical infrastructure, support services, costs and labor availability. J *et al.* (2016) pointed out that top level management and stakeholders from humanitarian organizations highlight the need of Define-Measure-Analyze-Improve and Control lack and

instability of resource (even human resources) within short time-frame operations. Authors also identified the necessity of assistance for front liners and implementing staff in terms of knowledge, quality information and flexibility from actual implementation tasks. These issues would allow field teams to be more focused on quality and accountability. These works show us that weakness of management and inefficient infrastructure are major issues. When the involved scenario calls for logistic assistance, the structuring of the humanitarian aid in the form of queues and service stations is necessary to streamline these processes.

To fill in these gaps, this work proposes an efficient management control as a method and low cost solution, which is structured by discrete-event simulation (DES) (BANKS *et al.*, 2010a) and Jackson networks (JN) (JACKSON, 1957) supported by $RFID$ and digital twin (DT) (VENKATAPATHY *et al.*, 2017a; ALAM; SADDIK, 2017; SCHLUSE *et al.*, 2018a). It is not within the scope of this work to study the specific $RFID/WEB$ tools, but rather to show that, as communication tools to support the use of DES, they can allow the real time monitoring of actual operations. The JN design is a choice due to the fact that it is a well-known topology for parallel and independent servers submitted to single-class products/services. In our study, we validated it for multi-class services (each server one different service). This method allows accurate and quick decision-making for planning and managing queues, personnel and materials for actual actions during short-time operations. The DES model validates the JN and results show its flexibility in following traffic flow variations.

The remainder of this paper is organized as follows: Section 2.2 discusses related work and some other applications. Section 2.3 describes the problem statement; Section 2.4 introduces the proposed simulation model. Section 2.5 addresses our remarks and discussion. Section 2.6 presents our conclusions and suggestions for future work.

2.2 Related Work and Background

There are many planning models for increasing throughput using JNs. Kim e Kim (2015a) is one example applied to medical emergency care with the use of hybrid priority with pooling for patients in risk of death. In contrast, our model allows the choice of one among many objectives (throughput, maximum time, average time) control submitted to $FIFO$ discipline. Most importantly, the method allows a real-time management control in a digital mirror supported by a simple and robust structure (JN) for multi-class services to care people in the state of vulnerability.

Another regular managing model is for optimal inventory allocation of multi-class products in single server systems, as studied by Cruz e Daduna (2017a). Differently, our

work is focused in DES for estimating the best results in a multiple parallel servers system applied to multi services in mixed queuing networks.

Bitran e Morabito (1996) show a optimization model for planning mixed queues networks with returning entities in a JN topology. Unlike, our work handle a DES real-time managing model for both operational and planning actions.

With regard to the DT use, normally there is the development of sophisticated models, e.g. the smart interaction controller for a digital twin tool offered by Alam e Saddik (2017) with fuzzy and Bayes supported by highly capable infrastructure. We propose a JN as a simple, low cost digital twin mirror application to be used in the presence of poor local infrastructure.

Venkatapathy *et al.* (2017a) propose the use of the dynamics of intra-logistics (associated to cyberphysical systems) with a system dynamics approach for the understanding of syntaxes that can lead to new services from inter-operating systems. Our focus is to provide functionality to the dynamics of intra-logistics to manage its emergent deviations.

Medical emergency procedures (entity is the patient, services are jobs, independent or not) is another example as showed by Xu *et al.* (2014a), can use $IoT-based$ methods where a chain of simultaneous procedures with responses coming through the Internet can guarantee patients' lives. However, besides a planning model our work also deals with a managing tool for operational decision making.

Some examples of practical processes illustrated by Fig. 2 as a description of a wide variety of scenarios with different types of operation, independents or not that can be executed simultaneously in different entities. A first actual example is a local process of loading trucks (entity is the truck, commodities are jobs) with the need to carry all product types as a whole cargo where commodities (corn, beans, rice, etc.) can be randomly laden in trucks by servers availability.

2.3 Problem Statement

This type of intra-logistics (GUNTHER *et al.*, 2008) operation assists people in need in an environment of multiple humanitarian services and they have to wait in very long queues to be cared for, or they must face intense competition for some assistance. People arrive at a supporting location ($Fig.2$) with a set of assistance needs from e.g. receptionists, nurses, medical doctors, dentists, and lawyers. Local infrastructure is usually poor; it needs material support and it also receives operational directions from volunteers that need to be guided too.

A people identification process is proposed for *Welcome & screening* assistance by the use of a regular passive *RFID* devices (for each individual) that can be completed with presence sensors control (e.g., *Arduino* based).

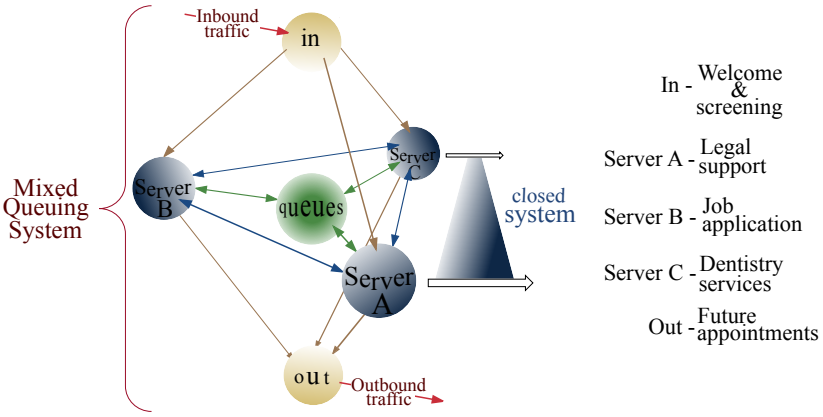


Figura 2 – General system model. A general view of a three servers system. It can be extended as n server system.

The arrivals and exits in any of the system *assistance points* (service stations) use *RFID* sensors to quickly identify quantities and time intervals which allows online supply of system statistics to support an intelligent system of a digital management interface. We assume that after an initial reception an individual already has a list of requested necessities. He/she has to hold for the next service in waiting rooms to be attended by availability to all the services. A person with no more necessities to be attended is evaluated and, in general, he/she is free to go away or to be back in a return schedule.

We ran a set of simulated experiments that assumes three different services for people that arrive on a daily basis, where the demand is organized and served by order of arrival. A full assistance has three services, each one with a service distribution time in a general design of one server for one node what can be easily extended for one node with many parallel servers. The servers simultaneously work in this class of large-scale processes, whose sequence of steps can be random, performed by a parallel network what often form long queues and cause loss and damage (ADAN; RESING, 2001; BITRAN; MORABITO, 1996; EECKHOUT, 2016a).

This is a class of problems of provision of large-scale services (it can also be transportation and/or transactions) in a multi-class process of queuing networks. It is a topology of Mixed Queues with a closed network as a core where entities are distributed by chance and dynamically for all servers without preemption (BITRAN; MORABITO, 1996).

The following are the assumptions of the proposed model:

1. server queues have no space constraints;
2. the scheduling policy is first-in-first-out (FIFO);
3. entities may randomly and equiprobably start from any free server;
4. both the arrival ($1/\lambda$) and the service time ($1/\mu_i$) of individuals/entities are Markovian;
5. the provision of services are independent from each other and there is no precedence relationships between individuals;
6. one server per service station, and each server has the same probability of receiving an entity;
7. each server provides a distinct service;
8. an entity from the inbound queue may only enter the system upon the exiting of another one, i.e. Under high traffic, the system operates under a statistical equilibrium. This means that an entity may only have access to a server once this same server releases another entity (BITRAN; MORABITO, 1996);.
9. each individual has to be served by a complete set of servers before leaving the system (Fig. 4 of Section 2.4) (i.e. the probability p of going from a server to the next available one is 100%);

The system with these nine characteristics may be modeled as JN. We designed a DES model (Subsection 2.4.2) which was analytically validated by JN and vice-versa. As a consequence, the system traffic logically behaves as a model of n serial queues, as confirmed by a comparison between the analytical and simulation models (Section 2.6). Once the model is validated, it is amenable to other distributions and system settings. We derive some simple equations (Subsection 2.4.1) for outbound traffic that analytically corroborates the simulation results and vice-versa.

In addition to monitoring processes, the proposed model may also be able to support planning and management of operations by dealing with relatively high traffic flow, in local operations or at distance, with limited number of available servers and very long queues. For example, as shown in Fig. 9 from Section 2.5, as the traffic increases, the JN model is able to correspondingly adjust its traffic behavior. Specifically, when one server overloads causing the formation of queues, the other servers can accommodate the excess load. Therefore, the capacity for load balancing is implicit in the JN model. This feature allows that the JN model may be used not only for static planning and dimensioning but also for online, dynamic management of an operation.

2.4 Modeling

Several value chains perform a similar basic flow outlined in the preceding section, (i.e. the sequential and undivided provision of services to an entity) that shows a response time with a given service time, based on previous knowledge/data of the system. For the particular case described in Fig. 2, the action conducted by the Office of the Secretary of Public Security of Rio de Janeiro City generated more than 13,000 assists to local citizens (OGLOBO, 2018a). The lack of official reports with detailed data justifies modeling with characteristic mean values. The general structure starts with a *Welcome screening* service, which assigns individuals to the adequate services. Clearly, there may be a large variability of the mean service times EB depending on the application, ranging from milliseconds (e.g. computer operating systems) to hours or even days/weeks (e.g. medical, military services).

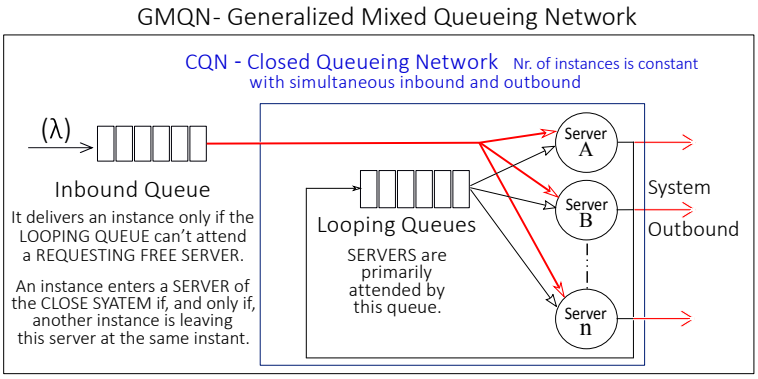


Figura 3 – System schematic model – *Server A* attends a job with $E[B_1]$ mean service time, *Server B* with $E[B_2]$ mean service time and *Server C* with $E[B_3]$ mean service time. Here, a server can be seen as a node with many parallel servers. *Looping Queue* is a logic queue that highlights the set of individual queues into the system core with closed queues network.

Figure 2 may be abstracted away as shown in Fig. 3, which shows the entry through the *Inbound queue*. This queue is used to implement an admission control that is not part of the JN model. If the system is bound to overloading, incoming entities are retained in this queue or even blocked away from entry in this queue if it is already full. Each server has its own queue and, the *Looping queue* is a logical representation for this set of server queues. As mentioned earlier, an entity may enter and leave this queue and remain in the loop until it has been serviced by each and all servers.

The model shown in Fig. 3 belongs to a specific class of mixed queuing systems with a closed queuing network in its core and n servers.

2.4.1 Simulation and Validation

The Jackson's theorem (JACKSON, 1957) states that the probability of an entity moving from one node to another is $p = 1$, as shown in the traffic model ($Fig.4$) where $\lambda = \lambda/3 + 2 \times \lambda/6 + \lambda/3$. The traffic in $Erlang$ (E) is described on a per-server basis ($\lambda/\mu_i < 1$ implies a stable system). Little's law (**??**) shows that the average residence time ($E[R]$) for each node is described as in Equation (2.1):

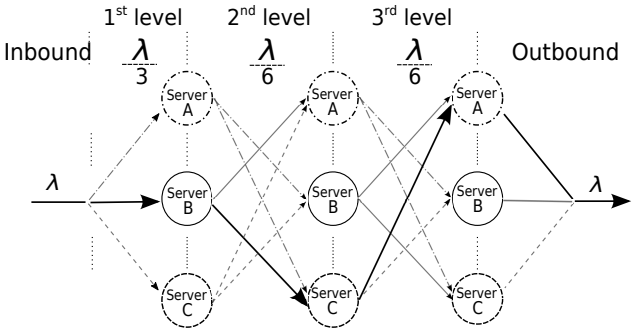


Figura 4 – Equivalent traffic model for three servers − *Server A* attends with $E[B_1]$, *Server B*, with $E[B_2]$ and *Server C*, with $E[B_3]$. Bold arrows show any equivalent traffic route is given by $inbound = 1^{st}level + 2^{nd}level + 3^{rd}level + outbound$.

$$E[R_i] = 1/(\mu_i - \lambda) \tag{2.1}$$

The mean residence time values in each node result in a traffic that may be calculated for each server, according to the traffic values related to the proportion of λ in each level as shown in Fig. 4 for three servers. In Fig. 5, a JN that is the logical equivalent of Fig. 4, an entity traverses three different jobs (three queues) with probability 1. Each server generates an inbound traffic to all other servers, i.e. as soon as an entity leaves a server there is 100% chance of this entity goes to another server, or leaves the system.

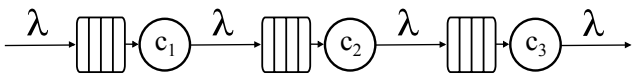


Figura 5 – Three single queues equivalent system. Servers can be set in any order, e.g., server C_1 provides service with E[B_1] ut, C_2, with E[B_2] ut and C_3, with E[B_3] ut.

For this reason, the arrival rate in each queue is the same as the overall arrival rate λ. This observation allows us to calculate the average residence time $E[R]$ in the system

through Equation (2.2):

$$E[R] = \sum_{i=1}^{n} E[R_i] = E[R_1] + E[R_2] + E[R_3] \tag{2.2}$$

It should be also noted that, from a performance viewpoint, an entity has to go through all servers and it does not matter which is the first one as it can be seen in Fig.5. As a first result we can use $E[R] = 105\ ut$ as a numerical example and, with Eq. (2.1), it is possible to calculate a system equivalent service time μ_{eq} through Eq. (2.3):

$$\mu_{eq} = \frac{1}{E[R]} + \lambda \tag{2.3}$$

$$E[B_{eq}] = \frac{1}{\mu_{eq}}$$

$$E[B_{eq}] = \frac{E[R]}{1 + \lambda\ .\ E[R]} \tag{2.4}$$

The use of Eq. (2.4) yields $E[B_{eq}] = 23.3\ ut$ for $\frac{1}{\lambda} = 30$. Otherwise,

$$\lim_{E[R] \to \infty} E[B_{eq}] = \lim_{E[R] \to \infty} \frac{E[R]}{1 + \lambda\ .\ E[R]} = \frac{1}{\lambda}$$

$$\mu_{eq} > \lambda \rightarrow E[B_{eq}] < \frac{1}{\lambda} \tag{2.5}$$

Equation (2.5) implies that there is an increasing number L_{eq} of entities in the system queues when the statistical equilibrium evolves to values of $1/\lambda$ smaller than system service times. To visualize it we can observe three quantities expressed by B_{eq} in Eq. (2.9); a quantity numerically calculated with results of the simulation model TEX_{limit} (Eq. 2.6) that shows the average exit time of outbound entities, or it denotes the actual average system response time considering lost entities; and TEX (Eq. 2.7) that reflects the possible system response time as a maximum due to it does not consider entities losses during high traffic (Figure 8).

$$TEX_{limit} = \frac{Replication\ time}{\#\ of\ outbound\ entities} \tag{2.6}$$

$$TEX = \frac{Replication\ time}{\#\ of\ created\ entities} \tag{2.7}$$

On the other hand, $E[R]$ can be written as in Eq. 2.8, the sum of an equivalent time parcel due to service, $Serv_{eq} = E[B_{eq}]$, and another waiting time due to queues, Q_{eq}, in two different intervals - under system limit and over system limit.

$$E[R] = E[B_{eq}] + Q_{eq} \tag{2.8}$$

We use Eq. (2.4) and Eq. (2.8) to derive Eq. (2.9) as a relation for $E[B_{eq}]$ and Eq. (2.10), for L_{eq} (equivalent mean number of entities in queue).

$$E[B_{eq}] = \frac{E[R]}{1 + \lambda \ . \ (E[B_{eq}] + Q_{eq})} \quad \rightarrow \quad E[B_{eq}] = \frac{E[R]}{1 + A_{eq} \ + \ L_{eq}} \tag{2.9}$$

First, in the general case for traffic flow under the limit, it results a calculation that must consider the servers in high traffic, but not overloaded (Eq. (2.10)). It means servers with the maximum offered traffic in a JN model is as $A_{eq} = \ \lambda \ \times (E[B_1] \ + \ E[B_2] \ + \cdots + \ E[B_n])$ $Erlangs$ what follows:

$$L_{eq} = (\frac{E[R]}{E[B_{eq}]} - 1) \ - \ A_{eq} \qquad If \ queue \ is \ stable, \frac{E[R]}{E[B_{eq}]} > (1 \ + \ A_{eq}). \tag{2.10}$$

Second, by the use of the Eq. (2.9) during overflow traffic, that means n $servers$ work with a full traffic condition (i.e. a crowded system with $A_{eq} = n \ Erlangs$ in the presence of $E[B_{eq}] = 1/\lambda$), the equivalent number of entities in queue $L_{eq_{hi}}$ can be written in terms of n as in Eq. (2.11),

$$L_{eq_{hi}} = \lambda \ . \ E[R] - (n + 1) \quad n = number \ of \ paralel \ servers \ (nodes) \tag{2.11}$$

By using some characteristic values as a numerical example (for three parallel servers), $E[B_1] = 1/\mu_1 = 20ut$, $E[B_2] = 1/\mu_2 = 15ut$ and $E[B_3] = 1/\mu_3 = 10ut$, we calculate:

$$\lambda = (1/30) \;\rightarrow\; E[R_{30}] = 105.00ut \;\rightarrow\; A_{eq_{30}} = (1/30) \times 45ut = 1.50 \;\rightarrow\; L_{eq_{30}} = 2.72$$

$$\lambda = (1/28) \;\rightarrow\; E[R_{28}] = 117.86ut \;\rightarrow\; A_{eq_{28}} = (1/28) \times 45ut = 1.61 \;\rightarrow\; L_{eq_{28}} = 3.40$$

$$\lambda = (1/26) \;\rightarrow\; E[R_{26}] = 138,37ut \;\rightarrow\; A_{eq_{26}} = (1/26) \times 45ut = 1.73 \;\rightarrow\; L_{eq_{26}} = 4.48$$

$$\lambda = (1/24) \;\rightarrow\; E[R_{24}] = 177.14ut \;\rightarrow\; A_{eq_{24}} = (1/24) \times 45ut = 1.88 \;\rightarrow\; L_{eq_{24}} = 6.50$$

$$\lambda = (1/22) \;\rightarrow\; E[R_{22}] = 285.48ut \;\rightarrow\; A_{eq_{22}} = (1/26) \times 45ut = 2.05 \;\rightarrow\; L_{eq_{22}} = 12.1$$

It is seen by Eq. 2.11 that as a consequence of $E[B_{eq}] > 1/\lambda$ be the limit, $L_{eq_{hi}}$ has no upper limit. The simulation experiments confirm results for traffic limit as can be seen next (Subsection 2.4.2).

2.4.2 Simulation Model Description

Running the simulation model built in a design of one single server per node figure 5 and setup for $FIFO$ policy (the logic of three servers in series), 100 replications of $100,000ut$ each, warm-up of $10,000ut$ and confidence level of 95%. Thus, in this way, other tests could be performed and some other results obtained after the simulation program be validated.

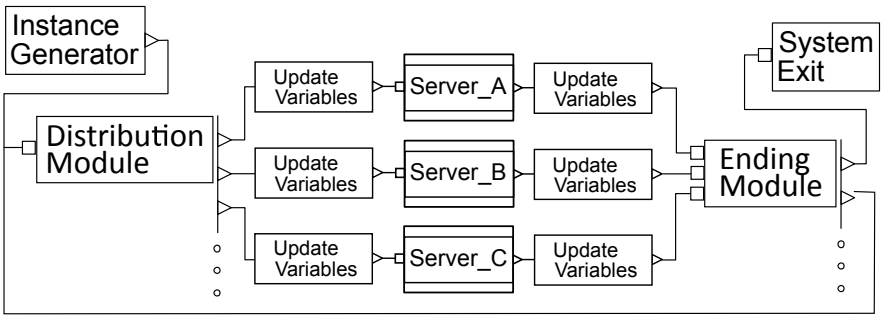


Figura 6 – Simulation model. *Distribution Module* draws equal probabilities to all servers driving them the requested entities. The block *Ending Module* returns unfinished entities or concludes finished processes.

The simulation model was developed with an entities generator that distributes both time between arrivals and respective service times and feeds the two-phase system: 1^{st}) it controls and limits the admission of entities with $FIFO$ policy; 2^{nd}) it distributes entities for servers. The whole system contains blocks *Update variables* that update data from/to entities and system what represent the inbound and outbound actions for the actual $RFID$ systems sensors.

In this example (*Fig.* 6) product or service demanded by the entity requires, as said before, a task performed in one of three different servers. Entity can leave the system only after it passed once by each of the servers. We describe in the next paragraphs two structural modules of the simulation model - *Distribution Module* and *Ending Module*.

Distribution Module (figure 6) - It is the double-staged admission control system. In the first stage it acts as a parking-lot (*System Limit*), where instances enter the system with $FIFO$ policy or leave it during overloading. It compares the number of system instances $varINLIMI$ with $varLIMIT$ variable (the total system capacity). If the value of $varINLIMI$ represents an under capacity system, the next queued instance enters the *Smart Buffer* that together with the *Probability Equalizer* makes up the second stage.

The *Smart Buffer* selects which servers have not yet rendered and sends the entity to the *Probability Equalizer*. This logic module is used to distribute traffic with equal probability for all servers that have not yet provided service to the entity. *Availability* blocks test if entity is already attended and if its routed server is free to receive it. It checks an availability variable named $varIN\_X$ (X is the server number) to open server access and test the attribute variable Ent_X to certify the correct entity to the free server. Released entities go to the *Avail* route and the denied ones go to the n/A route (not Available exit) to be returned to the *Probability Equalizer* module. Blocks *Update Variable* insert updated information to entities attributes (e.g., realized services) and system variables. This updating is even local or external (e.g., at distance) with *RFID/Internet* technologies. An important note is that in this configuration, buffers are queues that may guarantee independence from waiting entities and servers.

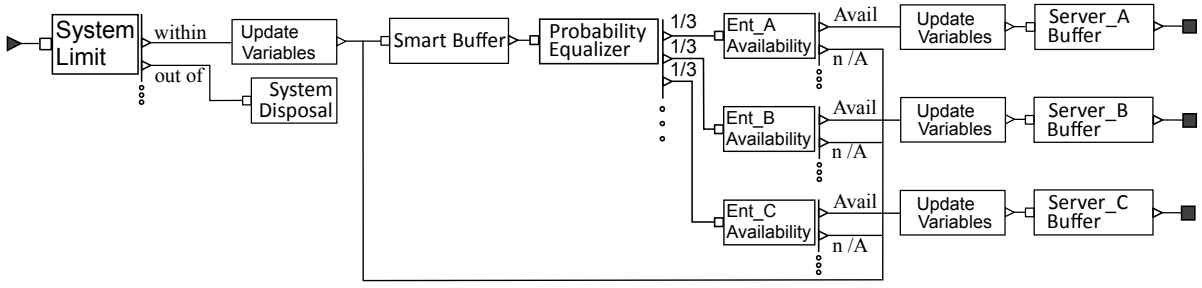


Figura 7 – *Distribution Module* schema. *System Limit* disposes undesirable overflow and the logical module *Probability Equalizer* guarantees the sum of probabilities is equal to one and the same probability to each of the servers. The servers buffers are independent queues that release entities by servers' request.

Ending Module represented in figure 7 is a router that returns unfinished instances, or instances with services to be done, to the *Probability Equalizer* and sends (*Return to servers* route) those instances without services to receive to system exit. It also counts finished

instances into 18 time-average intervals (the first 17 intervals distributed from $50ut$ to $4000ut$ and the last one to values higher than $4000ut$) to prepare the necessary data for a histogram that can represent the output probabilistic distribution.

First stage of *Ending Module* checks service completion of an entity by comparing values of entities attributes Ent_X (X is the entity type). Attended entities are identified to leave the system. The second stage of the *Ending Module* classifies outbound entities into 18 residence time intervals before exiting system.

For instance, starting simulations with inbound time average $1/\lambda = 30ut$, changing to $1/\lambda = 22ut$ and $1/\lambda < 21ut$, keeping the same other parameters the simulation model runs to return the next simulation mean-time averages:

$$for\ 1/\lambda = 30ut\ \rightarrow\ TAVG_{30} = 104.85ut\ \ with\ \ HW = 1.1278ut\ \rightarrow\ for\ E[R_{30}] = 105.00ut$$

$$for\ 1/\lambda = 28ut\ \rightarrow\ TAVG_{28} = 117.03ut\ \ with\ \ HW = 1.4580ut\ \rightarrow\ for\ E[R_{28}] = 117.86ut$$

$$for\ 1/\lambda = 26ut\ \rightarrow\ TAVG_{26} = 139.29ut\ \ with\ \ HW = 2.3726ut\ \rightarrow\ for\ E[R_{26}] = 138.37ut$$

$$for\ 1/\lambda = 24ut\ \rightarrow\ TAVG_{24} = 178.84ut\ \ with\ \ HW = 5.2408ut\ \rightarrow\ for\ E[R_{24}] = 177.14ut$$

$$for\ 1/\lambda = 22ut\ \rightarrow\ TAVG_{22} = 286.75ut\ \ with\ \ HW = 13.620ut\ \rightarrow\ for\ E[R_{22}] = 285.48ut$$

$$for\ 1/\lambda = 21ut\ \rightarrow\ TAVG_{21} = 475.96ut\ \ with\ \ HW = 37.159ut\ \rightarrow\ for\ E[R_{21}] = 491.59ut$$

The developed model is thereby validated and responds perfectly, but with low losses for $1/\lambda < 21ut$. In this case, model runs to $TAVG = 475.96ut$ and $HW = 37.159ut$ for the mathematical expectation of $E[R_i] = 491.59ut$ within the upper limit of the simulated range. The equivalent traffic model in $Fig.$ 3 adheres to the JN previously proposed. An important aspect of this work is that Jackson's analytical model validated the discrete simulation model and vice versa because there was difficulty in how best to represent the actual system. Once validated, the simulation program is ready for more results. The first one is the identification of the numeric system traffic limit. Assuming a λ range from 1/22 to 1/30 (this is regular range, in general, due to physical restrictions), the traffic $A = \lambda \times$ *average service time* in a stable condition, the upper limit for the most intense system traffic is calculated as 2.250 E ($\lambda = 1/22$). In actual operations these values must be obtained from actual system characteristics and constraints.

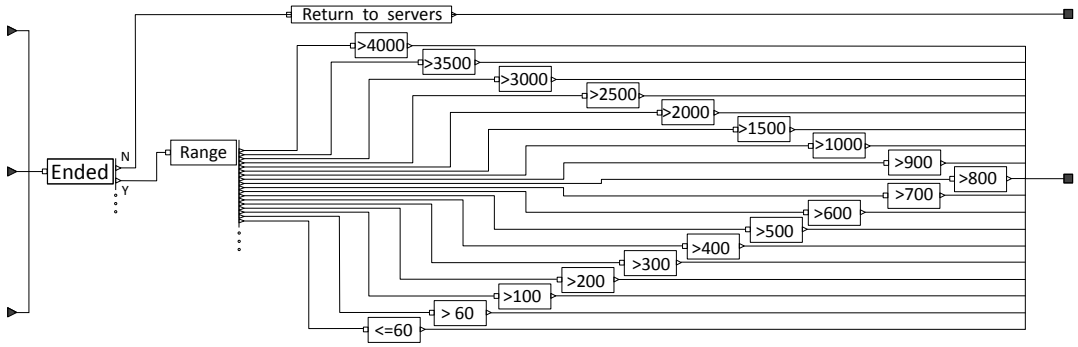


Figura 8 – *Ending Module* schema. The representation shows the upper line *Return to Servers* and the time range counters in *time units*.

2.5 Remarks and Discussion

Simulation experiments highlight system bottlenecks with the TEX_{limit} *vs* B_{eq} comparison as shown in Figure 8.

As λ increases over time, TEX_{limit} value tends to $E[B_{eq}] = 23.3ut$ what means that both tend to the highest service time of servers in system ($E[B_{highest}] = E[B_1] = 20ut$).

$$\lambda \rightarrow minimum,\ TEX_{limit}\ \ and\ \ E[B_{eq}]\ \rightarrow\ E[B_{highest}]$$

In an actual action with this *management tool* supported by a simple and small RFID/Internet communication structure, the management level can update the evolving data easily monitoring λ, all servers μ_i and estimate L_{eq} and TEX to make decisions, take actions and generate staff information and reports in real time.

This simple managing method goes beyond the example provided here. Servers perform simultaneous tasks in parallel but the system behaves as if it is in series. It happens due to the fact that this system performs different tasks to different entities at the same time for many time intervals. For this case study, we suppose independent tasks for three servers in parallel, but as the simulation model is validated, it can be also extended to tasks dependent on each other.

2.5.1 Implementation Strategy

The model discussed so far provides the theoretical framework for the scheduling of humanitarian aid across several parallel servers (or service stations). Clearly, the successful implementation of such strategy must be accompanied by modern technology that allows for the proper accounting for the process variables such as number of entities that enter the

system, time stamps, and residence time among others. Two technologies that may support the implementation is *digital twin* and *RFIDs* sensoring. Specifically, with the former, the collection of information system within physical operation feeds the mirror of a digital twin design that enables the automation of a high value-added chain (SCHLUSE *et al.*, 2018a).

Notice that it is not necessary for all levels and/or servers to be close together because, e.g., it is possible to manage these different operations remotely through a conventional Internet (GOYAL; FUSSELL, 2017). More recently, the support of IoT extended this perspective to meet new needs such as ubiquitous access, real-time analysis, availability to different platforms (Apple iOS, Google Android, Windows Phone) and spatial location (KUA *et al.*, 2017). As each instance (an individual) must pass through all the servers to complete the process, the distance between servers is not a hindrance. There are many simple $RFID$ designs available for collecting data and the Internet basis for transferring a small set of data is also available.

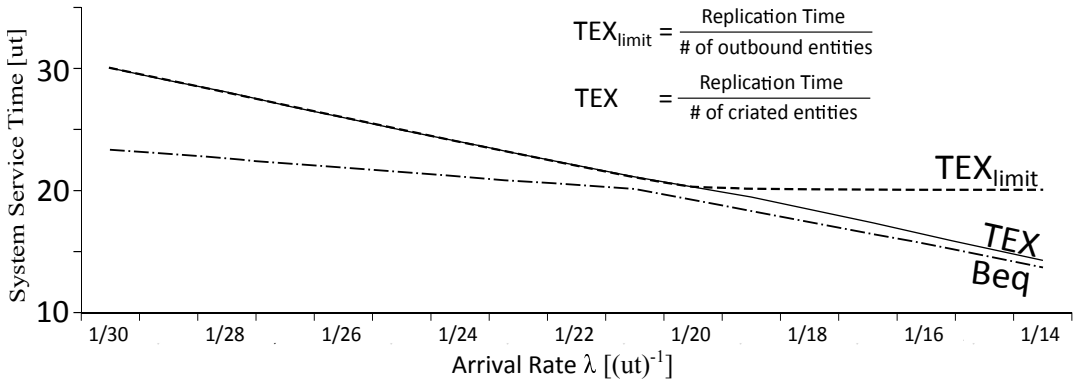


Figura 9 – System service time overview. TEX_{limit} shows the actual average system response time. TEX reflects the maximum system response time. Beq is the calculated value ($Eq.(2.9)$) for system response time.

2.6 Conclusion

When humanitarian aid scenarios strike, preparedness is of utmost importance, including how well response teams are able to react. This work has dealt with the issue of scheduling applied to managing humanitarian aid operations. A model based on a set of parallel servers and service lines using FIFO scheduling was proposed and validated by Jackson networks. The model has shown to be flexible in that is able to adapt to different traffic rates. It may be implemented as a management tool for both planning and operational support. Simulation design with buffers instead of regular queues allows the use of any scheduling policy. In an actual action supported by RFID/Internet, it can be planned a priori, and different scenarios and solutions can be tested and adjusted in real-time basis. Because the simulation

model allows the evaluation of the distribution of exit times, it is possible to monitor and interfere in the process to avoid users with extreme delays.

The Jackson network model is not just the basis for a simple and robust managing structure. It is also flexible due to its adaptive traffic characteristic. In the presence of high traffic, the system throughput tends to its maximum value. When a server is overloaded, its incoming traffic is redirected to the remaining servers, instead of being blocked on its queue. Flexibility is the consequence of the n-level buffer system, as the remaining servers work as a temporarily relief for the incoming traffic. Analyzing the simulation response curves, we clearly see from that they diverge around the slowest server service time ($20ut$ in the numerical example). As usual, this JN simulation model is a planning tool supported by the set of equations here addressed. The equivalent service time is easily accessed and fully adheres to the model mainly in high traffic. This JN queuing model is designed for exponential mean-time distributions (M/M/1) but the validated simulation model fits also for other general distributions, e.g., G/G/1 and the ones presented in Section 2.2, in a set of real world applications this network model can be addressed.

The DES characteristic model supported by $WEB/RFID$ technologies may also solve some operational issues highlighted in Section 2.1. Quick real-time answers can be delivered by the coordination/supervisory level (local or international, specific or multi-skilled) in terms of assistance to the operational teams (intra-logistics or inventory transportation) due to the soft and simple tool structure that can run new actual scenarios in few seconds. This characteristic shows that the distance between different steps that make up a complete process is no longer a problem (EECKHOUT, 2016a; FEKI *et al.*, 2013a).

As future work, we suggest studies to understand how the JN structure behaves with scheduling policies different from FIFO, for instance, using SJF (Shortest Job First) with known service times. Another possibility would be to focus in the effects of space restrictions to queues in the JN structure by using admission control. It is important to further study the effect of the increasing number of entities in the queue, and the use of measurements to find more realistic probability distributions, different from the exponential. Application of scheduling heuristics or analytical models at various decision points by using measurements from the studied environment can be other future work.

3 Identifying the SJF and FIFO Compromise

Flexibility in the accommodation of changes is a necessary step to meet business excellence and competitiveness. However, dealing with simple changes and instabilities, as e.g, changes in the scheduling policies and instabilities of the traffic intensity may turn regular daily operations into complex problems without a feasible solution. This work addresses the overcoming of response and performance limitations by the use of a method that combines discrete-event simulation with validated small-increments. The proposed method work on these kind of changes and instabilities that make, e.g., decision analytic models be out of the stated limits they were designed for. We investigate a previously designed and validated Jackson network system (FIFO) implemented to be operated through the SJF policy. This change is considered as a need that can drive better system performance. The implemented model proposes to synchronize a real-world logistic issue of laden multi-class products on trucks with the production process. The model comprises of a set of three parallel servers, where each server has its own service time to attend one unique class of product. Special attention was given to the compromise (positive, negative and null) between SJF and FIFO policies. The method shows to be simple, easy to implement and effective to return feasible responses. The method drives analysts to improve the system's throughput and to identify the relationship between the mean, the mean maximum and the standard deviation of residence times in a dynamic and complex system submitted to traffic instabilities.

Keywords: multi-class system, shortest-job-first, out of planning limit, incremental validation, performance analysis

3.1 Introduction

The need to achieve excellence and competitiveness requires quick and effective adaptation to new scenarios. Firms and institutions state requirements and parameters to reach objectives, to measure their effective performance and, with these needs changes can be understood in the sense of the trends that impose to a decision maker (or a stakeholder) to be able to distinguish between high and low performers (ANDRESEN; GRONAU, 2005). This paper addresses the issue of decision making related to complex systems in regular applications (e.g., supermarket's checkout lines, bank's teller lines, contact-center's lines, production's lines, passengers' boarding lines, etc). We present a queuing problem, with three servers (SANTOS, 2014) attending jobs in a Jackson network (JN) design (a special type

of parallel servers) under the shortest-job-first (SJF) discipline (not in FIFO). This study is relevant because of the models supporting analyses for decision making are performance limited (DEVORE, 2004; BANKS *et al.*, 2010b). Whereas, e.g., the system receives a requested change (or update), the whole system have to be evaluated. This means that the system's functionality submitted to changes could be out of the stated standards, e.g., because of the analytic model may be out of its boundaries. This paper discuss a practical approach supported by discrete-event simulation (DES) and small increments inserted to an initial and validated system.

The first objective is the identification of the variables' commitment between the first-in-first-out (FIFO) and SJF policies. The second, but not less important objective, is to show that this is a necessary method for the performance analysis and to keep the system's responsiveness of discrete-event complex systems not supported (or only partially supported) by measured data or by analytic models (as a result of changes, updates or instabilities).

A work's contribution is to show that there is a SJF policy compromise with the FIFO policy. This compromise can be seen through residence time variables, as e.g., the mean (TAVG), maximum mean (TMAX) and standard deviation of mean (TSTD). Other contribution is to address these compromise during the system changes from SJF to FIFO. Another valuable contribution is to emphasize that the use of small and validated increments, i.e. the incremental validation (InV) (IBRAHIM *et al.*, 2001; HAYNES; LENCH, 2003; HÄHNLE; MUSCHEVICI, 2016) is a simple and feasible solution. The emphasis in InV as a solution to address complex DES problems is a contribution itself and the use of InV to address answers to issues which analytic models can partially or, cannot work after changes or during instabilities, is also a contribution. A novel simple model based on Jackson networks, but attending jobs with the SJF policy (not FIFO policy) is another paper contribution. Within this context, this work attempts to answer the following questions:

1. Is InV easily implemented?
2. Could this simple InV solution address a DES complex problem?
3. Could InV support limited analytic models?
4. Is a Jackson's network design with SJF policy a feasible framework?
5. Can this kind of model be validated?
6. Are results comprehensive?

The remainder of this paper is organized as follows: Section 3.2 discusses theoretical backgrounds and related works; Section 3.3 states the problem, the method description,

introduces aspects of the incremental module with its main control function and addresses a brief description for our DES model pedagogically divided in four processes; Section 3.4 addresses the setup data for the incremented simulation model, results and discussion; Section 3.6 presents the conclusion and suggestions for future work.

3.2 Background and Related Work

We initially visit the qualitative and quantitative concepts of Incremental Validation and provide examples of their application.

(DEVORE; BERK, 2007) states that an approach is qualitative (or categorical) when it is focused in characteristics and values naturally structured by categories, as e.g., genre and school grade (male and eleventh grade). Differently, the author states that the quantitative approach is naturally centered in numeric values (e.g., age = 12 years, 7 months and four days, temperature = 112.7 kelvin).

In the context of software engineering applied to integration strategies and tests, (WHITE; LEUNG, 2002) and (ELSAFI *et al.*, 2015) argue that results must meet not only the functional validation, and also, the intermediate errors that are invisible to functional results due to the incremental changes. In this way, although the InV technique be widely adopted, one should expect limitations. An alert for the misuse of this technique can be found in (SPEAR *et al.*, 2006). Authors analyze different conflict detection strategies to promote a broad comparison of computational cost. Authors highlight that a process of successive and recurring InV may cause overvalued results. These increased values are especially high with the presence of high value jobs.

In the context of qualitative InV, (QUINTARELLI *et al.*, 2015) shows an algorithm that uses the consumer's new choice (the increment) to optimize a pattern (validation with a recognized context-aware model) that creates a list of suggestions. In other paper, an oriented graph of meta-model optimization, offered by (LAMOLLE *et al.*, 2015), identifies events (increments) from a sub-model to be context-validated with a well-defined and recognized model. (URSINI *et al.*, 2016) use an incremental method with DES to test the packet blocking in a single-server system validated with an analytic model. Increments are probability distributions individually inserted in a step by step basis and the results are validated.

(HÄHNLE; MUSCHEVICI, 2016) work with railway systems and consider changes and/or technologies implementations as increments and a simulation-based model as a virtual validated model. An increment is a step forward the original simulation model to decrease costs and improve usability/safety. Our proposal also uses incremental validation as these

ones, but neither in the perspective of adjusting entities into a set of qualitative attribute classes, nor in a context validation. Differently, we use a quantitative incremental validation approach to implement a new scheduling policy validated with a DES model to manage time-based variables.

In the context of the use of SJF and FIFO policies for the performance evaluation, (LEITE *et al.*, 2017) study radio frequency identification (RFID) tags and sensors that generate internet of things (IoT) traffic in AdHoc network systems. Authors analyze the output-CPU utilization through a validated simulation model and increments are different probabilities of connectivity during mobility. (KIM; KIM, 2015b) provides an evaluation method (for medical emergency care) based on DES and a hybrid scheduling of entities classes with a priority class policy and the FIFO class. (SANDMANN, 2006) uses DES to study the fairness performance in $M/G/1$ queues submitted to a hybrid SJF/FIFO scheduling that toggles one SJF entity and one FIFO entity. An evaluation method based on DES is studied by (CRUZ; DADUNA, 2017b) for optimal inventory allocation of multi-class entities in single server systems.

A hybrid method based on policy reorder plus numerical method plus DES is proposed by (HUM *et al.*, 2018) to evaluate responsiveness of supply chains in specific multi-stages and parallel topology considering possible backwards transitions. Authors consider responsiveness in terms of the probability of fulfilling customer orders within a quoted lead-time. (GROSOF *et al.*, 2018) give a performance analysis of multiserver queues submitted to the shortest processing time first policy (a variant of SJF) to compared it to other policies including FIFO. (BEKKER *et al.*, 2015) study a single server pooling system of many queues attended by SJF and others scheduling policies. Authors show that SJF's residence time distribution is associated to a generalized trapezoidal distribution. (COWDREY *et al.*, 2018) analyze the scheduling performance during queues of banking services. Authors identify the best performance with plain polices depends on the traffic and the institutions' requirements: SJF should be implemented during peak hours due to the best customer satisfaction and the most-profitable-job-first policy returns the highest institutional profit. (BOXMA; ZWART, 2007) study the performance of different scheduling that generates long tails distributions in relation to the optimality properties of the FIFO policy.

Unlike all of these papers, our method, which is based on DES with incremental validation, represents a multi-class system with the control of a specific number of SJF entities to generate analytic information from the SJF and FIFO commitment and, also, to analyze the multi-class system performance.

3.3 Proposed model

This is a logistics case of an actual local process of laden commodities on trucks for the food retail market. The trucks are entities, the orders of commodities are jobs, the service is the industrial process of laden one commodity of an order request on a truck, the conveyor-belts are servers and, the cargoes are sets of commodities carried by trucks. The trucks and the commodities arrive the system naturally and at random and, the quantity of a commodity is also random. The number of trucks and the total quantity of commodities fully attends the system. Each conveyor-belt attends to one unique type of commodity and this service cannot be stopped until it is finalized.

All the trucks are identical and, so are the conveyor-belts. The system processes three types of commodities, e.g., corn, bean, rice, etc., in one or more of their different categories. If the trucks are in a line, they are scheduled to be loaded with one of two predefined shipping sequence: the commodities are arranged in the order of arrival time in system or, in the order of the quantity of that commodity's type of the load in that specific queue. After a commodity is totally loaded, its truck moves immediately to another conveyor-belt to be shipped with one of the remainder commodities. A truck can only leave the system if its cargo is fully loaded, i.e., if its three commodities are boarded.

The initial model to the logistics case can be designed as in Figure 10 for scheduling policy FIFO (SANTOS *et al.*, 2018). Entities are independent and arrive in random (exponential distribution) to randomly access one of three identical servers.

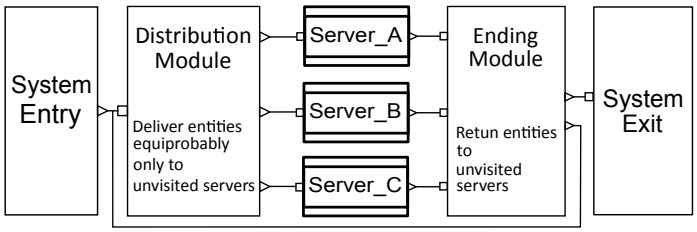


Figura 10 – A general view of the starting model for a three servers system described by (SANTOS *et al.*, 2018). It can be extended as n server system.

Each entity has a set of three independent job's classes. One class has its own mean and each class, its own and exclusive server (and vice-versa). All the servers work at the same time without preemption in a Jackson queuing network design (parallel basis). One full cargo carries three commodity types, which are randomly loaded according to the server's availability. Each server attends to one only truck at a time. Servers attend trucks only in FIFO scheduling policy declared at the beginning of the running process. As soon as a requested job is served, the entity is randomly delivered to a not visited server. The system

finishes an entity process when all of its jobs are served and then, the entity is ready to leave the system.

The FIFO scheduling supports the JN, differently from the SJF policy that is partially an open issue. Thus, SJF inserts limitations to the JN model and so does to the applied system's analysis as mentioned in Section 3.1. This is an applicable issue for the InV technique that submits a controllable small change (e.g., the SJF policy) to a known and validated model as in Santos *et al.* (2018), to obtaining feasible results, as for example, the mean maximum time in system. Here are examples of other increments handled by the InV technique: Leite *et al.* (2017) present a single extension of the system topology that is changed from five to 20 nodes system design; Quintarelli *et al.* (2015) show a change of one single parameter of a typical system process where a new consumer choice changes his consumption profile; Ursini *et al.* (2016) incremented a new probability distribution. The system began with exponential distribution and, it was changed to weibull;

3.3.1 The SJF/FIFO incremental model

The logistics case allows us to find out if the system's responses after submitted to incremental changes are (or are not) within the expected results, e.g., the system mean residence time and its variance and system flow. Experiments would also allow us to observe aspects of other variables such as, e.g., the service time. The activity diagram in Fig. 11 shows that before an entity enters a server, the system's logic verifies the policy rule to be applied. It means that the system is able to work under a dynamic change of mode driven by the new incremental module.

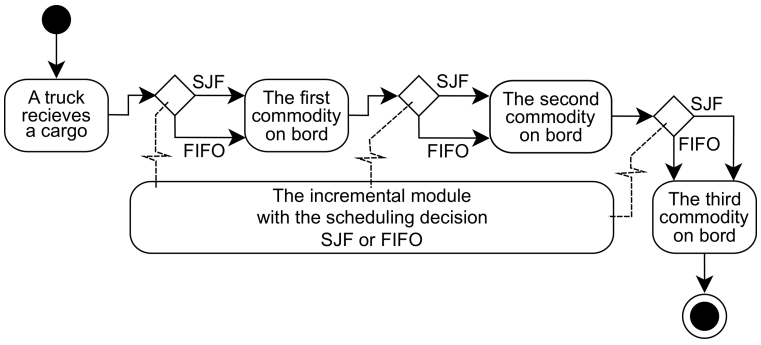


Figura 11 – Activity diagram for the incremental validation model

The choice for the SJF policy is because of the following considerations: the job size is identified at the time the entity enters the system; the simplicity of the SJF policy; this policy is easily applied to real life; SJF is broadly known as performing the shortest mean

residence time from the non-preemptive policies; even though, SJF's analytic model is still partially an open issue.

The system complexity with the SJF implementation (Fig. 12) implies the model service time distribution is changed from the original FIFO's exponential. It can be expected a long-tail service distribution because it is no longer a time dependent distribution changed to a size dependent one and, thus, the JN model no longer represents the system. Our motivation is to reach one step forward the known limits of this kind of complex queuing model. Entities move to a server through a policy as stated by the block *Policy* in Fig. 12. This block executes the policy set out by the system policy control (for instance, SJF).

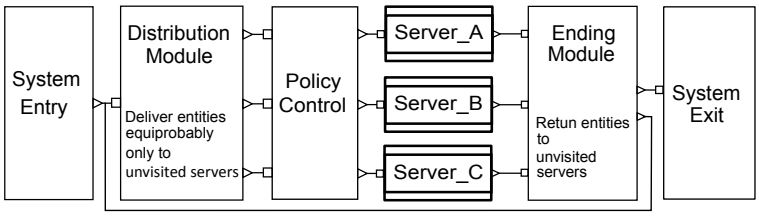
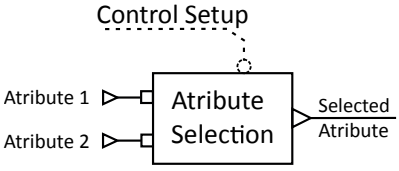


Figura 12 – Incremented system model. A general view of a three servers system with new *Policy* incremented modules. It can be extended as n server system.

InV allows complex characteristics of actual systems to be added as small incremented steps to guarantee the evolution of an accepted and validated model (HÄHNLE; MUSCHEVICI, 2016; HAYNES; LENCH, 2003). To illustrate the implementation, Figure 13 represents the conditional logic design. It sets value *zero* for the $FIFO$ scheduling or value *one* for the SJF policy. Values are recorded in a specific attribute (entity variable) $atrVALUE$ according to a system setup variable $varINPOL$ set at the beginning of the running process.



Control Setup	Selected Atribute
0	Arrival Time
1	Service Time

Figura 13 – Incremented module schema of the policy control applied to each server.

Servers are parameterized to look for the shortest value of $atrVALUE$ from the entities in the server buffer. A server reads this value that sets the processing time of each entity - the shortest arriving time for FIFO, or the shortest processing time (entity's size) for SJF.

3.3.2 A general description of the Simulation model

It is not our purpose to give exhaustive details about the simulation model, its variables and relations that lead to the actual system similarities. This model description gives a short and necessary overview for the implementation of the incremented DES model. The model considers the SJF policy and the most important variables that may lead to the expected functionalities (to decrease the TAVG values and to facilitate the study of the maximum mean residence time in system (TMAX) behavior). This model can be implemented and executed in most of the commercial software (e.g., *Arena* ® (as we did it) or *ProModel*®) in their *Student Versions*.

The simulation model was developed with an entities' generator that distributes both time-between arrivals (λ rate) and the service times (μ_i) for the system represented in Fig. 14.

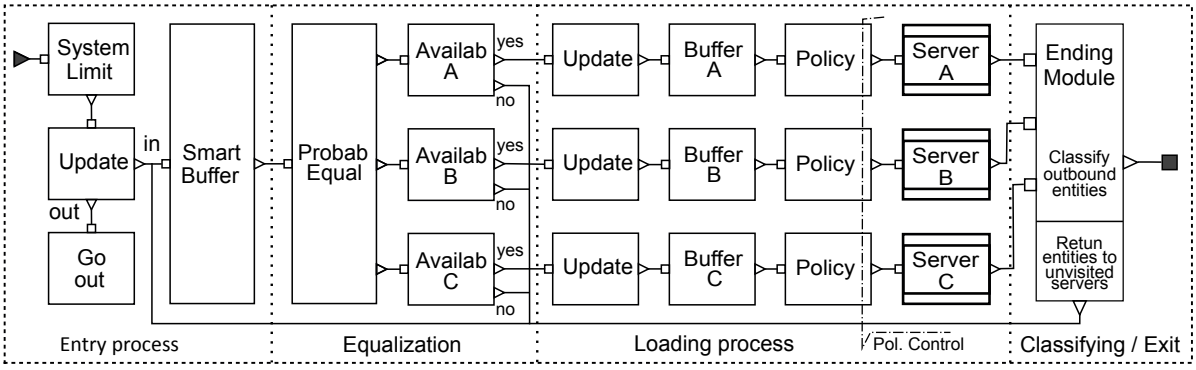


Figura 14 – Simulation Module schema. *Entry process* disposes undesirable overflow. *Equalization* guarantees equiprobabilities to every server. The *Loading process* imitates the charging process onto trucks and the *Classifying/Exit* represents the reentry process and it also ranks entities into classes.

This design allows the use of increments with other distribution probabilities as exemplified in Section 3.1 to eventually change the implemented exponential distribution. Figure 14 is organized in five logic sections - *Entry process*, *Equalization*, *Loading process* and *Classifying/Exit*. The *Entry process* logic controls the total number of entities to represent the actual system capacity or software limitation. A system requirement is recorded by the setup Variable *varINLIMIT* to be compared with the updated total number of entities in system. This is the key to let an entity enter the system or, to dispose it if it is required. The Smart Buffer identifies which are the buffers an entity need to be served and the available buffers in the system stating the necessary parity. The *Equalization* process distributes entities to only one of their unattended servers with equal probability. The *Smart Buffer* identifies the free servers to deliver entities to the *Probability Equalizer* that distributes and guaran-

tees traffic with equal probability to unattended servers. *Avail* blocks check an availability variable named $varIN\_X$ (X is the server number) to open a server's buffer access.

The whole system contains blocks *Update* that interchange data from/to entities, resources and the environment. The system exchange information, e.g., the Sever_B is free for the entity, or the entity is unattended by Server_A and Server_C. In a general way, they represent inbound and outbound actuators for any system information with an automation initiative as, e.g., *RFID/Digital Twin* systems devices (SCHLUSE *et al.*, 2018b; XU, 2012).

The *Loading Process* actually represents the process of products being loaded on board of trucks (in general applications, it is the serving process). Entering the *Loading Process*, an entity waits to be attended in a server's Buffer that frees it according to the scheduling policy defined in the block *Policy* by the *Policy Control* requirements. This routine allows an entity to be effectively attended by the *Server* with the amount of service stated by the job-sized of that entity. The *Classifying/Exit* process outlines two tasks. A router that returns instances with not-provided services to the *Equalization* process and also, the *Classifying/Exit* logic sends the finished entities to be ranked into values. It allows the necessary data analysis of histograms that can represent output probabilistic distributions (e.g., number of entities per classes of sojourn times). Entities finally exit system.

3.4 Case Study

The initial FIFO-policy model (Fig. 10) is incremented with the new SJF scheduling policy. The analysts expect a specific system response for the SJF-incremented system. Once those expected results are confirmed (mean residence time shorter than FIFO, changes to the service distribution, long tail residence time distribution and increase in the TMAX values), the incremented model may be deemed valid. Those results must be further validated against real data (even if it is a small data-set).

3.4.1 Parameters

It is important to highlight that we use ***ut*** as the acronym of *time unit*. It is done to avoid the misuse of the symbol *TU* that means *transportation unit* in a typical logistic case (JANIĆ, 2014) represented by this study (Table 2).

The DES experiments used 100 replications of 100,000 ut each, warm-up of 10,000 ut, the 95% confidence level and characteristic values as a numerical example for three parallel servers: inbound average time $1/\lambda = 30\ ut$ and the expected service time of each server $E[B_i] = 1/\mu_i$, and so, $E[B_A] = 20\ ut$, $E[B_B] = 15\ ut$ and $E[B_C] = 10\ ut$. We decrease the

Tabela 2 – Incremental validation parameters to the new simulation model

Replication parameters		Traffic parameters (ut)	
Number of replications	100 units	Inter-arrival time ($1/\lambda$)	30
Replication time	100,000 ut	Service time A ($B_A = 1/\mu_A$)	20
Warm-up time	10,000 ut	Service time B ($B_B = 1/\mu_B$)	15
Confidence level	95 %	Service time B ($B_C = 1/\mu_C$)	10
Total system capacity	145 entities		

inbound average time in a *1 ut* basis to reach $1/\lambda = 21$ *ut*, keeping the same other parameters.

3.4.2 Results of the first incremental model

The simulations ran the first incremented system to provide the new simulation residence times in order to be compared with FIFO as in (SANTOS *et al.*, 2018) and the new results driven by the SJF changings (Table 3):

Tabela 3 – Observed values of variables subjected to the incremented model with the SJF policy

$Traffic$ (Erlang)	$TAVG_{FIFO}$	HW_{FIFO}	$TAVG_{SJF}$	HW_{SJF}	$TSTD_{FIFO}$	$TSTD_{SJF}$	$E[R]$
	Time Units (ut)						
1.5000	104.85	1.1278	83.206	0.49186	67.982	66.256	105.00
1.6071	117.03	1.4580	88.666	0.57657	76.876	77.029	117.86
1.7307	139.29	2.3726	96.743	0.80984	94.312	98,807	138.37
1.875	178.84	5.2408	108.73	1.1421	124.40	141.82	177.14
2.0454	286.75	13.620	135.01	2.8119	206.99	288.73	285.48
2,1429	475.96	37.159	172.78	6.9341	318.51	617.97	491.59

Simulations with the SJF increment model in FIFO scheduling repeated the results in (SANTOS *et al.*, 2018) where the calculated mean residence time $E[R]$ is totally within the confidence interval represented by $TAVG_{FIFO} \pm HW_{FIFO}$. The SJF policy returned expected reductions of mean-time values $TAVG_{SJF} \pm HW_{SJF}$ (compared with FIFO) as seen in Table 3 and Fig. 15. The data show an improvement in the system's TAVG. There is reduction of the average-time's values of the jobs in those FIFO's tail classes, but it introduces new and higher values of tail's classes in SJF (Fig. 16, also related to the greater values of standard deviations $TSTD_{[}SJF]$ compared to the FIFO's.

Whereas FIFO returned TMAX values shorter than 600 ut and 2565 entities with residence time less than 175 ut, SJF reach TMAX of 2200 ut with 2816 entities up to 175ut of residence time. FIFO returns 85.76% of total flow within the reange of 175 ut and SJF,

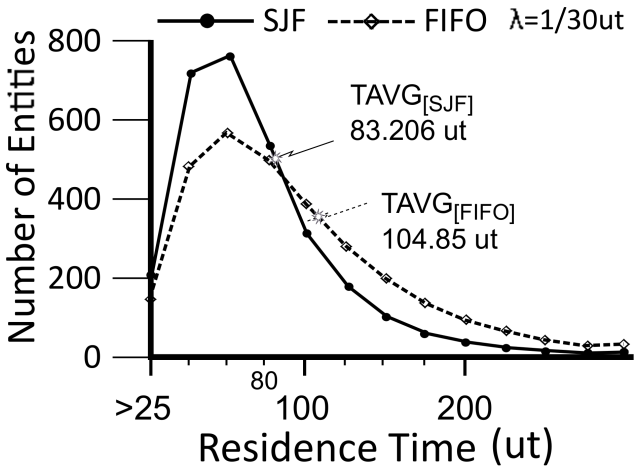


Figura 15 – Mean simulated residence times for FIFO and SJF in the mean residence time distribution for SJF and $FIFO$.

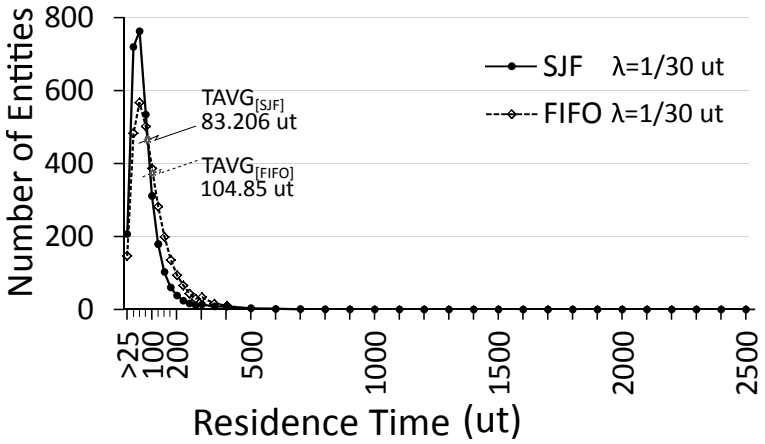


Figura 16 – Number of outbound entities. Distribution of mean number of entities subjected to SJF and $FIFO$.

93.77%. This increase means the SJF improved the flow within the 0-175 ut class in 9.786% compared to FIFO. The SJF job's class of 175-600 ut (that corresponds to the FIFO's tail) resulted 57% less entities. There were less than 0.2% (= five) entities in the SJF's tail with values greater than 500 ut and, only 4% of total entities, i.e. 126 entities with residence times greater than 175 ut. Incremental validation made other variables available, as e.g., SJF standard deviation $TSTD_S$ and, also confirmed that SJF allows the smallest values of TAVG (please, see (SANDMANN, 2013) for M/G/1 analytic results).

3.4.3 Further analysis through the incremented model

The analysis of the percentage of total flow with the SJF incremental model (Fig.17) shows an increase in the number of released entities in the shortest range (0-100 ut) by almost

15% what is greater than the 10% of the range 0-175 ut discussed in the later section.

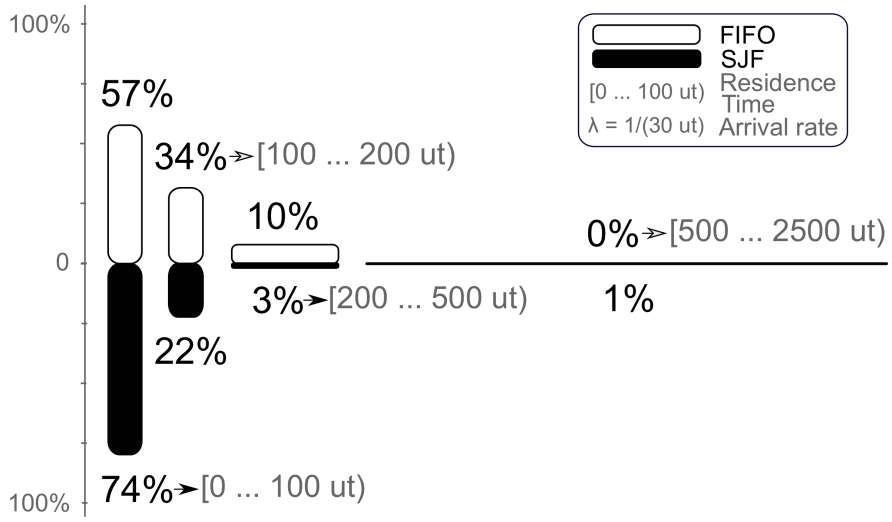


Figura 17 – Outbound entities per classes of residence time. Highlighting the outbound entities flow in three classes: 0 - 100 ut, 100 - 200 ut and greater than 200 ut for SJF and FIFO .

As one can observe in Figs. 16 and 17, the distribution of the SJF's residence times is an adjustment applied to the FIFO's one. The shorter SJF's mean residence time (from FIFO's 104.85 ut to SJF's 83.206 ut) happens together with a shorter half-width (from FIFO's 1.1278 ut to SJF's 0.49186 ut). This fact identifies a higher density of jobs around the mean residence time of the SJF policy. As a consequence of the Kleinrock's conservation law that, in simple words states the total system delay remains the same no matter the scheduling policy (KLEINROCK, 1965), the decrease of the number of entities in the classes with high residence time is compatible with the sense that the farther away a job's class is from the mean residence time, the less entities belongs to the class (Fig. 18). As a confirmation of this sense, whereas the number of entities increased in the class of shortest SJF's residence time values, the number of entities in the highest ranges ($>$ 200 ut) decreased from 10% within the FIFO range to 4% in the SJF ranges and the extreme high values of TAVG ($>$ 600 ut) are less than 0.5% of the total number of jobs leaving the system.

Other feasible observation of the SJF model is the changes of $TMAX_S$ (Fig. 18). The SJF's $TMAX_S$ variable value submitted to the highest traffic value suffers an intense increase to about 18,000 ut if compared to FIFO's $TMAX_F$ (1450 ut). InV also alerts to the necessity of study if SJF changes the type of TMAX distribution compared to FIFO. We do start this study comparing SJF and FIFO in different traffic intensities as it can be seen in the Section 3.5.2.

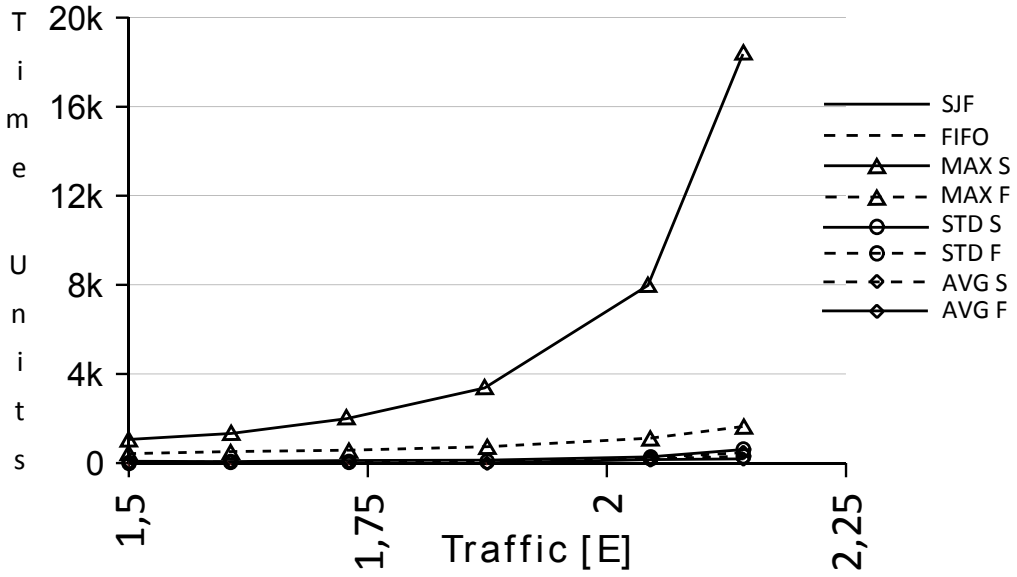


Figura 18 – Outbound variables. Highlighting the behavior of the outbound variables with traffic intensity for SJF and $FIFO$.

3.5 The SJF and FIFO compromise

To understand the policies compromise, we wish to observe how the system behaves as long as it changes from one policy to the other. To do so, we use new increment modules that allow us to handle both policies in a same running cycle. In a step-by-step process, we insert a controlled number of jobs of the SJF policy from zero jobs to the full capacity number of jobs in an initially FIFO policy system. In a progressive procedure, the number of FIFO policy jobs decreases from the full capacity of jobs to zero jobs in system, i.e. to reach a full SJF policy system. We observe, measure and analyze the TAVG, the TMAX and the TSTD values of each experiment step to discuss the SJF and FIFO compromise.

3.5.1 A general view of the hybrid policy simulation model

It is not the purpose of this work to deal with the details of the hybrid simulation model (Fig.19), but only to tackle it in a general overview to provide understandings of the new increment functionalities. The most important model functionality is the Hybrid Policy Control that adds both policies in a same process cycle. A changing policy functionality is also of the same relevance. The third new functionality is an occupation/vacancy module that gives flexibility to insert jobs of a policy in any point of the running process and, additionally, it allows elasticity to the amount of jobs of each insertion.

It is important to notice that any non-preemptive policy starts in FIFO-type regime until there are no waiting lines in the system. It means that a policy different from FIFO,

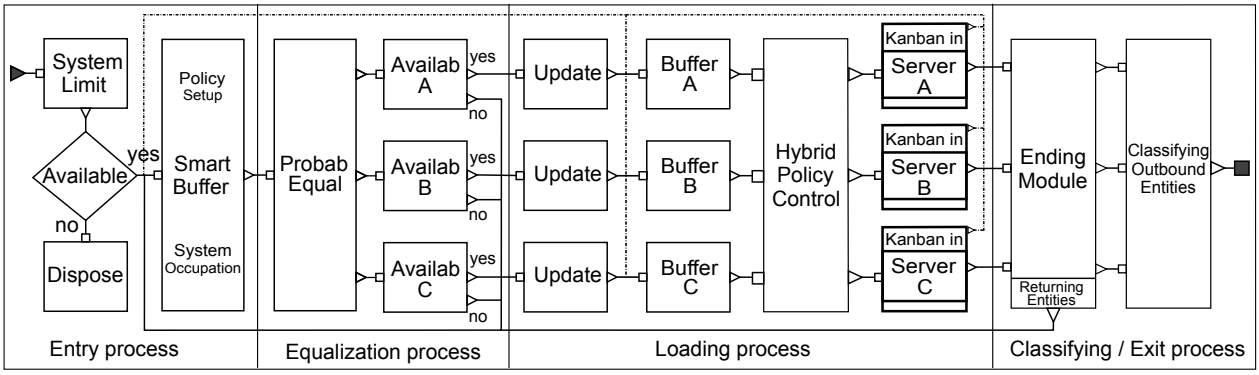


Figura 19 – Hybrid Policy Simulation Model. System Limit disposes undesirable overflow. Probability Equalizer guarantees equiprobabilities to every server. The servers buffers are independent queues and attend servers in a Kanban design. The Hybrid Policy Control manage both SJF and FIFO policies.

e.g. the SJF only exists in the presence of queues. The simulation experiments driven by the model represented by Fig.19 start with system set up in SJF policy and as soon as the total number of jobs in system reaches a given number S, the policy changes to FIFO. Whereas the system reaches the S number of entities (e.g., 20 entities) the set-up policy changes from SJF to FIFO what imposes the system SJF policy to operate with no more than S entities (20 entities in this example). The occupation/vacancy module of the Entry process identifies if the number of entities is increasing (e.g., from 10 to 11 to 12 ... 20) to change in exactly 20 entities during the occupation time. During the vacancy time the module does similarly when (using the same example) the number of entities comes down from 25 to 24 to 23 ... 20 and the system changes again, but now from FIFO to SJF policy. The simulation continues changing back and forth to reach its stationary state. To ensure that the server meets only the current policy a job's request is sent to the server's queue as quick as the server is free. This is the Kanban procedure (OHNO, 2009) incremented in the model.

3.5.2 Results of the hybrid model

The experiments consider the system full capacity of 145 entities. Results in Fig.20 show responses for the whole system capacity each five-entities step to observe TAVG, TMAX and TSTD subjected to the traffic intensities of 2.5000, 2.3684, 2.2500 and 2.1429 Erlang (E). The system goes from the full quantity of FIFO entities to the full quantity of SJF entities (visually from the left to the right side of each graph). This step by step procedure allows us to identify the variable values in relation to the number of SJF entities in a stationary system, i.e, the SJF compromise with FIFO.

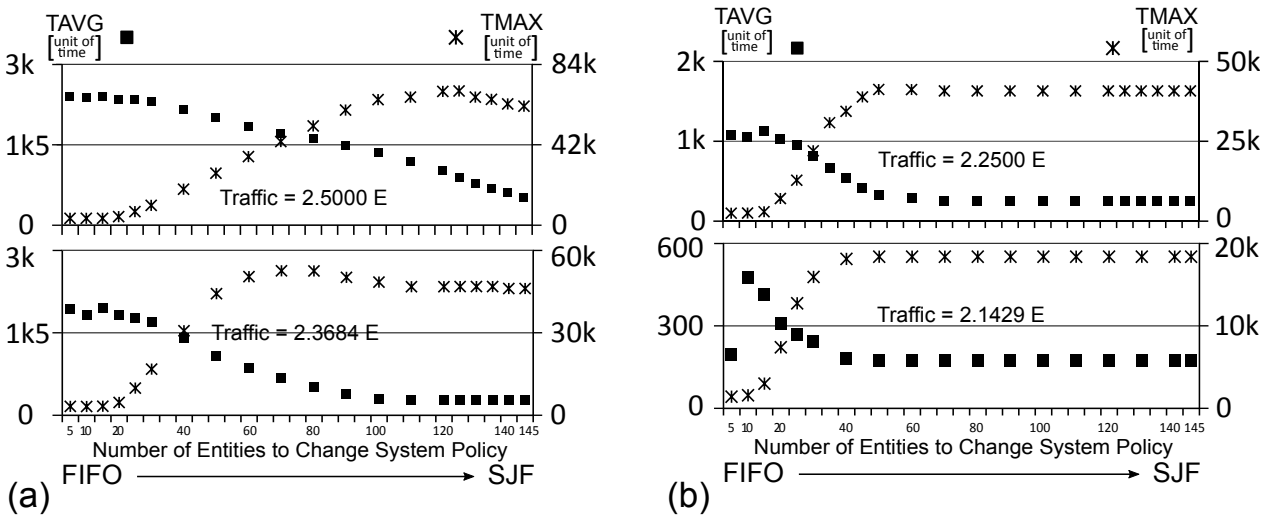


Figura 20 – The average and maximum residence times. The residence times versus the number of entities in system with different traffic intensities for SJF and $FIFO$.

The mean average values TAVG and the mean maximum average values TMAX are shown to have a region of inverse relationship, i.e. a region of trade-off (Fig. 20). Whereas FIFO evolves to SJF (from the left to right side of the graphs), one can observe this region where TAVG values decrease and TMAX increase. This is true until the TMAX variable reaches its peak value and from this point both TMAX and TAVG decrease to their own steady-state value.

Using the same kind of observation, TSTD and TMAX are shown to be in direct relationship in Figure 21 until the TMAX peak value and both the variables (TSTD and TMAX) decrease to the their steady-state value.

TAVG, TMAX and TSTD repeat their mutual compromise for all traffic intensities. It is important to note that the system submitted to the highest traffic intensities (e.g., 2.3684 E and 2.5000 E) the higher TMAX and TSTD values happens before the system reaches its total system capacity of entities. For example, the graphic Traffic = 2.3684 E in Figure 21(a), these highest values are within the range of 50 to 100 SJF's entities.

3.5.3 Results of the classified responses

The Classifying Outbound Entities Module is able to offer some understanding of the variables behavior. During FIFO policy, Fig. 22 shows that TAVG of all the traffic intensities happens within the region the greater concentration of entities (more than 90% of the total entities). The low traffic (FIFO 30), the medium traffic (FIFO 22), the high traffic (FIFO 20) and the extreme high traffic (FIFO 14) reveal this fact.

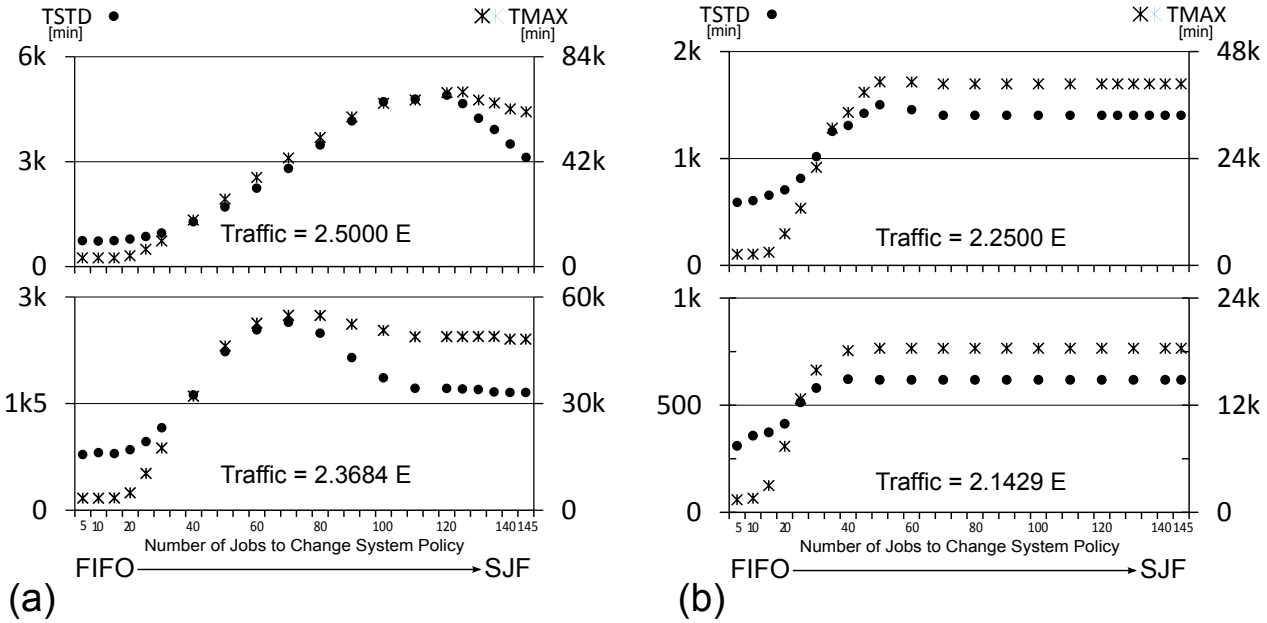


Figura 21 – The maximum mean residence times and the deviation of the mean residence time. The time values versus the number of entities in system with different traffic intensities for SJF and $FIFO$.

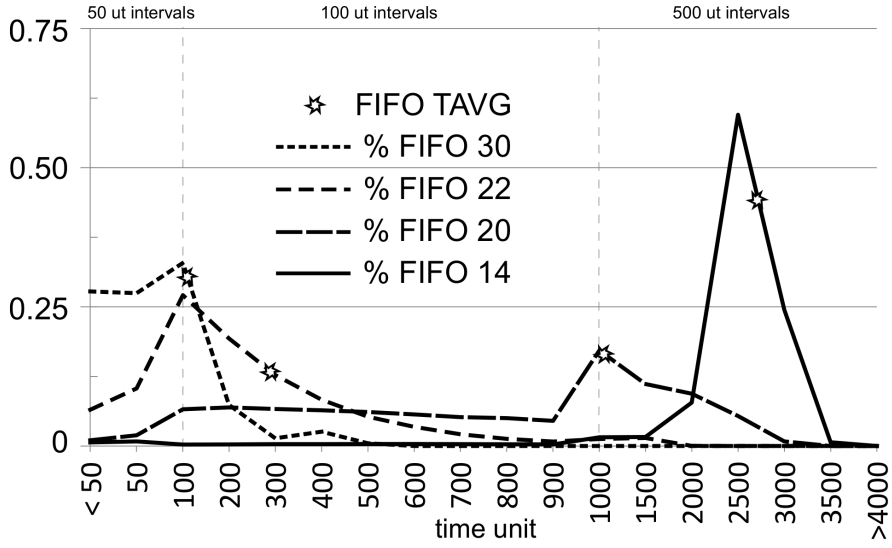


Figura 22 – Relates TAVG and the distribution of entities in % of total entities per different traffic intensities during FIFO policy. Note: Because of the broad range of times, the horizontal axis has three different sized-intervals: 50 ut (at left side); 100 ut (at center); and 500 ut (at right side).

Differently from FIFO, during SJF policy, Fig. 23 shows that the greater concentration of entities (more than 90% of total) are within the lower classes of TAVG (smaller than 200 time units) for all the traffic intensities - low, medium, high and extreme high. The TAVG values for low and medium traffic are within the most populated ranges, but the high traffic (SJF 20) is out of this region. The TAVG of the extreme high traffic is far from this region, overcoming (in this case study) 1000 time units. Managing the small amount of entities of the SJF's tail is feasible due it is fewer than 10% of total entities. It is reasonable to expect a decrease of TAVG and TMAX (after managing jobs in the distribution´s tail), and an increase in the system throughput as consequences of the Kleinrock's conservation law. The system can, e.g., divert these jobs traffic to an auxiliary server system. This action put the total delay down and with the conservation law allow the system to fill this decrease with more traffic of shorter time jobs. This possibility cannot be applied to FIFO policy.

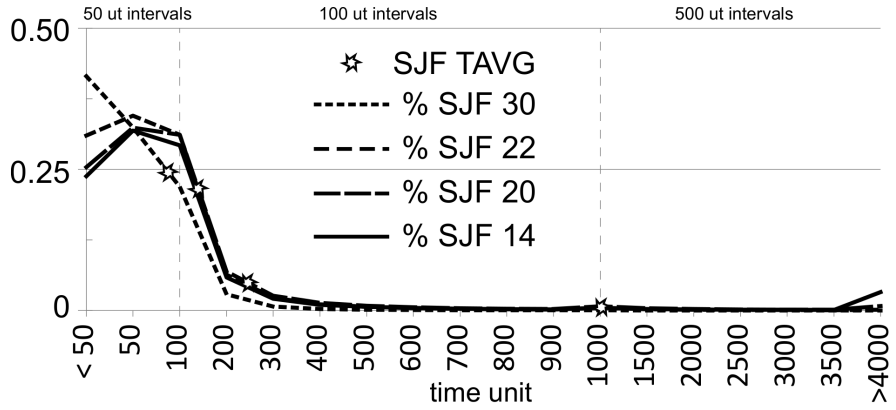


Figura 23 – Relates TAVG and the distribution of entities in % of total entities per different traffic intensities during SJF policy. Note: Because of the broad range of times, the horizontal axis has three different sized-intervals: 50 ut (at left side); 100 ut (at center); and 500 ut (at right side).

Figures 20 and 21 show that during the increase of job's quantities, a system submitted to the SJF policy has the highest TMAX values appearing in the intermediate quantities of jobs (e.g., 50 to 100 entities with 2.3684 Erlang traffic), but more than 90% of jobs in the shortest TAVG classes (Figure 23), with the average times in steady-state lower than the highest values.

Two issues involving the SJF policy must be brought to attention. The first one is that the extra high TAVG values could not be the most important impact factor in reducing the system performance. Notice that, during extreme high traffic, these longest job's values (the right side of the Figure 23) are the last to be processed under the SJF policy, but a plain SJF system (right side of Figs. 20 and 21) have lower values of TAVG, TMAX and TSTD than the highest reached values. Secondly, as a consequence of the first issue, we need

to further investigate the relationship between the quantity of jobs in a class, and the impact of that class on a system's performance submitted to the SJF policy.

3.6 Summary and Conclusion

We showed that through Incremental Validation we could obtain the results for the performance analysis of discrete-event complex systems which are not supported by measured data or by known analytic models. The experiments showed the implementation simplicity with a kind of one-step-forward methodology that reach complex systems where responses are initially not fully predictable. This Incremental Validation simplicity associated to general expected results for the additional characteristic allow easy DES validation, as shown with the SJF's residence times and its long tail response distribution.

Unknown responses become feasible and actually observed with the use of Incremental Validation despite of the limits of the analytic models. This was showed with the example of the maximum mean residence time TMAX in a simple graphic through the traffic, or through the entities flow analysis studied per residence time classes. The relationship between variables submitted to different policies can be also observed. SJF is compromised with FIFO as shown in the inverse relationship between TAVG and TMAX and the direct relationship between TMAX and TSTD independent of the traffic intensity.

Incremental validation clearly shows the differences of models' performances and allows the feasibility of models derived from known ones as in this case of a JN model changed to fit, e.g., the SJF policy regardless of modeling formalism. Differences from FIFO and SJF are clarified. Whereas the FIFO's mean residence times are always inserted in the range of times with the highest concentration of jobs, the SJF policy does it only in low and medium traffic. In high and extreme high traffics the mean residence times is positioned in the tail's curve out of the highest populated region of jobs. This is another commitment from FIFO and SJF policies, what qualifies the SJF scheduling to manage the extreme high residence times. It is important to remember that incremental validation is limited to processes that do not cause its successive and recurring usage. This kind of application may cause overvalued results maximized by the presence of jobs with extreme values.

We would suggest the incremental validation to study how the concentration of jobs in a class of time average can impact the system performance. Another investigation would be how multiple possible trajectories (LEYE *et al.*, 2014), as e.g., to represent different traffic intensities can be managed to drive a system into better solutions.

We are ready to continue with studies, supported by incremental validation and DES,

on the trade-offs of SJF variables. Experiments need to go further on the design of other traffic intensities with the same, or associated to other policies. Another interesting future work would be the use of regression to design a mathematical behavior of a specific variable, as e.g., the SJF maximum mean time.

4 A New Framework for the Performance Analysis of the Single-Server Non-preemptive Scheduling under Varying Traffic Conditions

Paper submitted to: The Journal of Scheduling (JOSH), Springer/Nature

In this work, we propose a new framework (model, methodology and implementation) for the performance analysis of a single-server, non-preemptive policy under the shortest-job-first scheduling and varying traffic conditions. A case study is presented where we show the applicability of the framework through the analysis of the mean residence time of a job, when the input traffic intensity is increased from low to high traffic. The implementation of the framework is based on a discrete event simulation model extended by a continuous simulation model through a common interface. The framework is general in the sense that it is not limited to the analysis of the mean residence time of a job, as other performance indicators may be also included in the analysis. Through the application of the framework, and under the assumption of a dynamic and variable input traffic, it is possible to find the size of a job x that ensures a quality of service requirement such as the mean residence time, i.e. as long as the long-term scheduler of the framework only admits jobs of size less than (or equal to) x into the system.

keywords: modes of operation; mode changes; Shortest Job First; scheduling; dynamic loads

4.1 Introduction

Research on scheduling and performance analysis has advanced quite significantly over the years and offered new solutions, techniques and tools for static scenarios and conditions. However, the real-world lies at the other end of the spectrum, as it may be characterized by quite dynamic conditions, e.g. including varying traffic intensities, workloads, and unpredictable faults just to name a few. Thus, the need for scheduling and performance analysis that embrace unpredictable changes may be deemed to be one of the most current and realistic requirements in this field. Given that the goal of scheduling is to achieve an optimized

behavior of some resource-limited system, the value of a scheduling and analysis approach usually depends upon its continuing relevance to the current mode of operation (or set of states). A mode of operation is broadly a design abstraction that captures a given set of environmental conditions, and requirements, which are met by a specific system behavior that is implemented by a single schedule (i.e. set of jobs) (SANTOS *et al.*, 2018). For each mode, a different and custom schedule is required, along with its performance analysis, one that addresses the specific requirements of the system at that particular phase of operation. Once the conditions surrounding the system (or internal to the system) are changed, a mode change ensues and a new schedule is required and loaded, one that meets the new constraints ((PEDRO; BURNS, 1998), (MARTINS; BURNS, 2008), (PEDRO, 1999)).

Within this context, this work tackles the performance analysis of jobs in a single-server non-preemptive system, where the size of a job is *a priori* known and traffic is variable and unpredictable across a number of modes of operation (e.g. mode A for low, B for medium and C for high traffic, as shown in Table 4). The policy considered in this work is mainly the shortest-job-first (*SJF*), although first-in-first-out (FIFO), last-in-first-out (LIFO) and random order (RO) were also analyzed for the sake of comparison. In this work, the major performance parameter under analysis is the system mean residence time (TAVG), and its role is to satisfy a user-specified quality of service (QoS) requirement.

Therefore, this work represents a step towards tackling the performance analysis of jobs scheduled under unpredictable changes of the traffic intensity (i.e. job arrival rate) while maintaining a given QoS requirement. More specifically, the key question being asked is *"given a changing, variable traffic intensity (Erlang [E]) in the system, what is the size x of a job that ensures a certain QoS requirement?´´* For example, consider an application that needs to keep the job´s mean residence time bounded to a maximum value in order to fulfill its QoS. We wish to find out what is the value of job size that reduces TMAX to a level below a given value (i.e. a setpoint defined by an application) for a certain traffic intensity. Once the value of the job size x is known, there are a number of strategies that may be employed to ensure the QoS requirement. Clearly, one possible action would be to shorten the average service time in a server by preventing jobs of a size larger than x from entering the server, e.g. by diverting the larger jobs to a secondary (or alternate) server or by temporarily blocking them. As the traffic level changes and surpasses a given threshold (thus implying a new mode), a new value of x may be found that allows the system to comply with its QoS. The answer to this key question is mainly shown in Section 4.3, where Table 5 indicates the value of x (i.e. job size) that maintains the mean residence time at the indicated level. For example, if only jobs of size *x=644 s* or less are kept in the system for the traffic intensity at 0.45 E, then it is possible to ensure that the TAVG is limited to *289-290 s* and, TMAX is

limited to *2141 s*, thus preserving the desired QoS level.

To this end, we developed a framework that combines a continuous (CSM) with a discrete-event simulations (DES) that answers this class of questions. It allows the performance analysis of a system under instabilities (i.e. under dynamic changes such as varying traffic), where the analytic and the discrete models in isolation are not able to provide the desired performance analysis due to the involved complexity.

The model allows a designer to reason about and relate QoS to a job size under specific traffic intensity. To illustrate the applicability of the framework, we present a case study that analyzes the regular but not trivial problem of a single server system scheduled by the SJF policy with variable (increasing) traffic.

The remainder of this paper is organized as follows: section 4.2 discusses related work, section 4.3 introduces the proposed framework with the continuous simulation and discrete-event simulation models and their integration interface. section 4.4 offers a simulation case study showcasing the applicability of the framework along with the results. section 4.5 presents the conclusion and future work.Appendix X shows that the SJF has the shortest mean among non-preemptive means; Appendix X presents the analytic models for the mean and variance of the non-preemptive disciplines. Appendix X shows the implementation details and Appendix X offers some guidelines collected through the experience in this work.

4.2 Related Work

The SJF policy, its applications and performance analysis have been emphasized in the literature in the last decade. An overview of recent developments is presented in this stion.

Tang *et al.* (2013) proposed a dynamic scheduling algorithm for super-computing application to prioritize jobs to resources allocation by the use of an adaptive scheduling based on workload characteristics. It is a variant algorithm from a hybrid FIFO/SJF policy that schedules batches instead of units of jobs. Differently, our framework is a general integrated application of DES and CSM simulations to identify a job-size that satisfies QoS requirements across system instabilities (e.g. traffic's variation).

Xoxa *et al.* (2014) discuss the FIFO and SJF disciplines to emphasize that the SJF discipline allows a shorter average time and lead the system to excessive residence times. Our work is not limited to these two disciplines but also discusses other non-preemptive disciplines, such as LIFO and RO. In addition, our approach seeks to evaluate the queuing behavior in the presence of instabilities of traffic and also, the issue of performance considering fairness.

A *handoff* performance is addressed by Owaidat *et al.* (2015) on mobile communication systems considering *the remaining time in the cell policy*, where the mobile station in greater velocity should *handoff* sooner, what resembles a kind of SJF scheduling. Our work aims to evaluate the predictability of the residence time of a job of a certain size (in relation to the first and second moments), submitted to SJF for general applications.

Wierman *et al.* (2007) analyze the first and second moments of the SJF's residence time according to a parameter named the expected maximum duration depending on the service length. We associate this approach to the collective approach of discrete simulation results, in terms of system mean values applied to actual situations. In a different way, Sandmann (2013) addresses the fairness issue regarding service disciplines. Our work is also related to this issue, but we analyze its limits depending on the offered traffic to be extended to any traffic distribution.

Elmougy *et al.* (2017) study system design issues applied to cloud computing infrastructures and application services. They designed a hybrid algorithm named SJF-RR-Dynamic-Quantum, SRDQ to minimize the starvation dilemma in complex computing and big-scale. Differently our issue is focused on a simulation solution related to general problems, for example, as in services, manufacturing, production or transportation processes. Our study proposes continuous simulation associated to discrete-event simulation to take advantage of the fairness metric. It is proposed to understand the unknown behavior of the system submitted to traffic instabilities during the SJF policy looking for improvements of the expected results of regular delay metrics.

Guda *et al.* (2016) analyze a variant of the non-preemptive SJF scheduling policy named shortest-variance-first (SVF). The objective of minimizing the sum of the expected total system duration time addresses the issue of total room occupation cost for single surgical room procedures. SVF considers jobs in a normal convex distribution, known expected job length, known mean and variance of the mean in an organized sequence. Differently, our work analyses the system performance of the pure SJF where systems variations and instabilities lead to partially unknown delays. Our proposal associates the advantages of the job's parameters (e.g., its service length) and the general system behavior (e.g., the system residence time) under unknown, variable traffic conditions, in order to improve queuing system performance.

The work by Wierman e Harchol (2005) is the closest to ours in the scheduling literature. They provide an analytic model that evaluates the mean average time as a function of the job-size x. However, unlike our work, their

model does not consider a *variable (i.e. with fluctuations) traffic intensity*. Clearly,

the inclusion of a variable traffic charge in a model is crucial since, in most systems, traffic varies significantly during normal and abnormal modes of operation. To our knowledge, no work in the literature evaluates a global performance indicator as a function of both the job size x and the dynamic traffic intensity (modeled in a step-by-step basis) at the same time in order to meet a quality of service requirement as shown in the next Section.

4.3 Proposed Model

The framework (Fig. 24) combines DES and CSM to maintain a given QoS requirement in the presence of changes or instabilities as presented in section 4.1. Given a non-preemptive discipline (e.g. SJF), the model allows the (user) specification of one or more requirements, e.g. that TMAX satisfies a certain bound ($\text{TMAX}^{REQ} < \text{VALUE}$). It is best described in terms of inputs, outputs and structure, as follows:

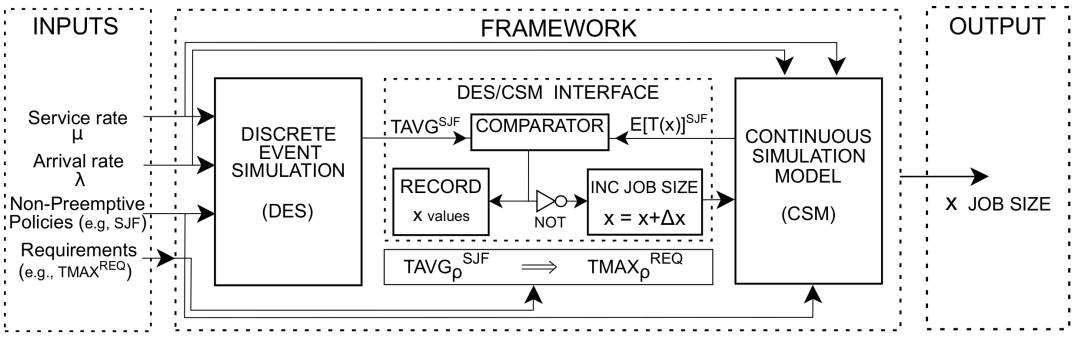


Figura 24 – Overview of the Proposed Framework. Symbol REQ means *requirement* and $\rho = \frac{\lambda}{\mu}$. The INTERFACE can be fed with the comparative variable the system requirement needs the simulators to deliver, not only TAVG^{SJF} and E[T(x)]^{SJF}.

Inputs: The general input data (Fig. 24) are the arrival and the service distributions (λ and μ respectively), which are both exponential probability distributions. As shown in section 4.4, we keep the service rate μ at constant value while increasing λ (in a step-by-step basis) in order to evaluate a given discipline;

Output: The final output is the x value, which represents the job size that meets the user-specified system requirements, e.g. the reduction of TMAX_{DES}^{SJF}. The framework considers not only the input set, but also TMAX, the fairness related to the predictability of x job-sizes (known a priori or when it can be estimated) in SJF ((MOR, 2010)) and the system submitted to traffic instabilities. Whereas the requirements and inputs may vary (see discussion in Section 4.4.1), we adopt TMAX as the desired system goal to be achieved by the framework.

Goal: Specifically, the goal of the framework is to analyze three classes of performance attributes (both mean and variance) for SJF and FIFO: 1) The service delays; 2) the system delays, and 3) the mean system delays as a function of the job size.

Operation: The DES model evaluates the mean system performance to use one of its identified variables as a reference value to an interface module. This interface compares collective (TAVG_{DES}^{SJF} from the DES) to individual $E[T(x)]^{SJF}$ from the CSM) indicators (Fig. 24). The simulation may stop as the $E[T(x)]$ approaches TAVG^{SJF} within a pre-specified margin/tolerance. At this point, the value of x is recorded and it represents the job size that direct or indirectly satisfies the set requirement. The general operation of the framework is described in Section 4.3.3. The major blocks are next described.

4.3.1 Discrete event simulation model (DES)

Fig. 25 shows the discrete event simulation model, which is essentially an M/G/1 single queue characterized by the waiting queue (W) and a service provider that is modeled using a service distribution H. It was modeled and analyzed using a discrete event simulator ($Arena^{®}$) to obtain regular system time-delay indicators for SJF. Other disciplines such as FIFO, LIFO and RO were also analyzed.

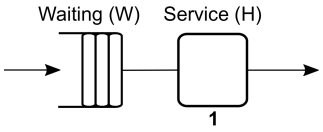


Figura 25 – Discrete-event simulation schematic model.

Clearly, it is possible to generate a number of performance indicators from the DES model. The selected output of the DES module is the $TAVG_{DES}^{SJF}$ since it is the one used in our case study (Section 4.4) to demonstrate the approach. It is also the variable that is used to analyze the requirement set in the case study ($TMAX < TMAX^{REQ}$). All the framework inputs are inputs to the DES module, the exception being the user requirement. The requirement is a terminating condition to the simulation and it is associated to TAVG_{DES}^{SJF}.

4.3.2 Continuous simulation model (CSM)

The CSM block is shown in Fig. 26. It implements and solves (WIERMAN; HARCHOL, 2005) equations, which define the mean residence time (E[T(x)]) for SJF as a function of the traffic generated by the jobs with size less than or equal to x. The authors' predictabi-

lity analysis evaluates both FIFO and SJF disciplines. Their predictability criterion is given by Equation (4.1):

$$\frac{Var[T(x)]}{x} \leq \frac{\lambda E[H^2]}{(1-\rho)^3} \tag{4.1}$$

This equation represents the capacity of consistently identifying different job sizes in a queuing system, given the variance of the job with size x $Var[T(x)]$ along the value of x. It is applied to fairness by scaling the service instabilities $\lambda E[H^2]$ through the system idleness $(1-\rho)^3$. The mean residence time $E[T]^{SJF}$ of x is given by Equation (4.2):

$$E[T]^{SJF} = E[H] + \frac{\lambda E[H^2]}{2(1-\rho(x))}$$

$$E[T]^{SJF} = \frac{\rho(x)E[H]}{2(1-\rho(x))}\left[1 + \frac{\sigma^2(H)}{\{E[H]\}^2}\right] \tag{4.2}$$

where H is the service probability distribution (e.g. exponential), $\rho(x)$ is the traffic generated by jobs with duration x, $E[H]$, $E[H^2]$ and $\sigma^2(H)$ are their first and second moments and the variance of H, respectively.

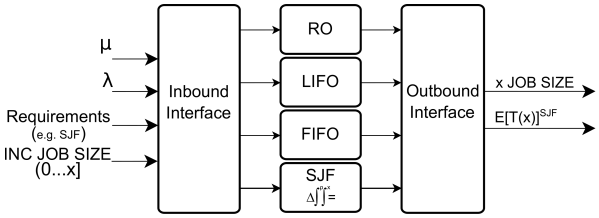


Figura 26 – Continuous simulation schematic model (CSM).

Thus, the inputs to the CSM block are the arrival distribution represented by (λ), the service distribution H represented by (μ), and the x job's size, in order to evaluate the traffic $\rho(x)$ and then, the final block output, i.e. the mean residence time ($E[T(x)]$). For example, when $x = 100$, the equation integrates all values of job sizes from 0 up to 100 to generate $E[T(100)]$.

The specific details of the implementation of this module are illustrated in Appendix C. The time dependent equations (FIFO, LIFO and RO) integrate the arrival distribution from 0 to ∞ to return the system mean value. The SJF equations integrate the service distribution (H dependent on x job's duration) from 0 up to x by evaluating the integrals of the first and second moments of H (i.e. $E[H]$, $E[H^2]$). Thus, we have means and variance values for a specific x value, which can vary depending on the goal of the application. We

can use *Matlab*®-*Simulink*® resources to evaluate the dynamic behavior of the SJF, i.e. its mean and variance (Appendix C, Fig. MG1).

4.3.3 DES/CSM interface

The key element of the DES/CSM interface is the Comparator. It checks if the CSM mean residence time – $E[T(x)]^{SJF}$ – lies within the DES mean residence time $TAVG_{DES}^{SJF}$ interval – i.e. $E[T(x)]^{SJF} \in (TAVG_{DES}^{SJF} \pm \Delta T)$. If so, then its output enables and records the x value. Otherwise, this signal enables the increment of x. Then the x value (from the INC JOB SIZE) enters CSM to evaluate the analytic model and results $E[T(x)]^{SJF}$.

We choose the DES system residence time during SJF – $(TAVG_{DES}^{SJF})$ – as a reference to feed the predictability equations of the CSM and reach the required job size x. In this approach, only the DES time average variable – $TAVG_{DES}^{SJF}$ – is observed by the Comparator. It is important to highlight that this is just one choice of variable among a set of possibilities. For example, another choice could be TAVG^{SJF} $\pm$ 1 *Standard Deviation* (or any other exiting variable), depending on the actual application, parameters and requirements.

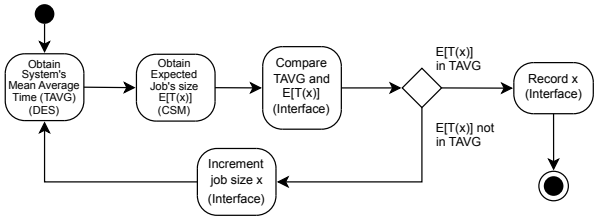


Figura 27 – Activity diagram of the Proposed Framework.

Figure 27 shows the flow of operation of the model (activity diagram).

4.4 Case Study

Having described the model and its constituents in the preceding section, in this section we exercise the framework to generate Tables 5 and 6 from the parameters described in Table 4. The goal of the case study is to analyze the following variables:

1. Mean service and residence times;
2. Variance of the service and residence times, and
3. Predictability.

In essence, we vary the traffic and for each intensity we find out the x value that imply $E[T(x)]^{SJF}$ within the stated $TAVG_{DES}^{SJF}$ interval.

4.4.1 Parameters

The following parameters were used to configure the case study:

Service: The mean typical call (E[H]) is 180 s with service variance of $\sigma^2[H] = 180^2 = 32400\ s^2$. The H distribution is considered as exponential; *Arrivals:* Calls arrives at a constant λ rate (for each simulation). The inter-arrivals distribution is considered as exponential. *Traffic load:* The basic testing model is a simplified telecoms switching system. The system is configured around three modes of operation depending on the traffic intensity. Mode A represents low-traffic (from 0.050 to 0.450 E), mode B models medium traffic (from 0.700 to 0.800 E); mode C is the high traffic (from 0.9 to 0.92 E). Note that the thresholds that differentiate from low to medium and high traffic are application dependent and in this work they are arbitrarily defined for illustration purposes. The traffic values ($A = \rho = \lambda \cdot E[H]$ E with $\lambda/\mu < 1$) are (Table 4): 0.180, 0.257, 0.450, 0.700, 0.750 , 0.800, 0.900, 0.910 and 0.920 E.

Tabela 4 – General Simulation parameters.

Mode of operation	*Traffic (A)* Erlang	*Arrival time* Second	λ
A - Low Traffic	0.050	3600.00	0.0003
	0.180	1000.00	0.0010
	0.257	700.39	0.0014
	0.450	400.00	0.0025
B - Medium Traffic	0.700	257.14	0.0039
	0.750	240.00	0.0042
	0.800	225.00	0.0044
C - High Traffic	0.900	200.00	0.0050
	0.910	197.80	0.0051
	0.920	195.65	0.0051

Simulation runtime: We carried out 10 replications of $1 \times 10^5 s$ simulation run for statistical significance of 95%.

4.4.2 Results for the DES

This section provides analyses for the maximum and mean residence times, its variance as functions of the increasing traffic to SJF and $FIFO$.

The DES performs the mean variance in the queue ($\sigma^2[W]$) and the variance of the system residence time ($\sigma^2[T]$). The variance of the service time is $\sigma^2[H] = 180^2 = 32400$ s^2 (exponential distribution). Therefore, Eqs in Appendix *Analytic Model* from this paper *New Framework*, allow us to understand how $\sigma^2[H]$ behaves during the experiments. Figure 28 shows a significant change in the SJF service distribution variance compared to FIFO (similar traffic values up to 0.3 E).

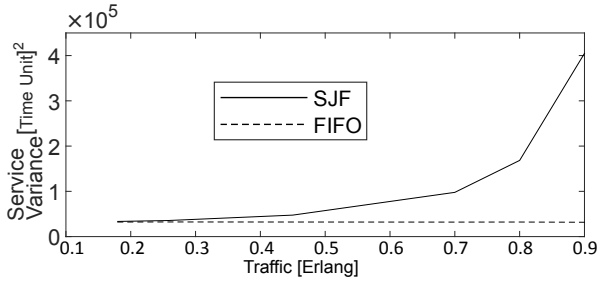


Figura 28 – Variance of the service distribution.

The effect of the SJF discipline changes the exponential distribution of the service times ($E[H] = 180s \rightarrow Var(H) = 32400\ s^2$), and its variance sharply rises up with the increasing traffic intensity. The service time distribution is no longer an exponential distribution and SJF exceeds all other disciplines' variances. On the other hand, all disciplines (FIFO, LIFO and RO) keep the service variance close to $32400s^2$ (dashed line in Fig. 28).

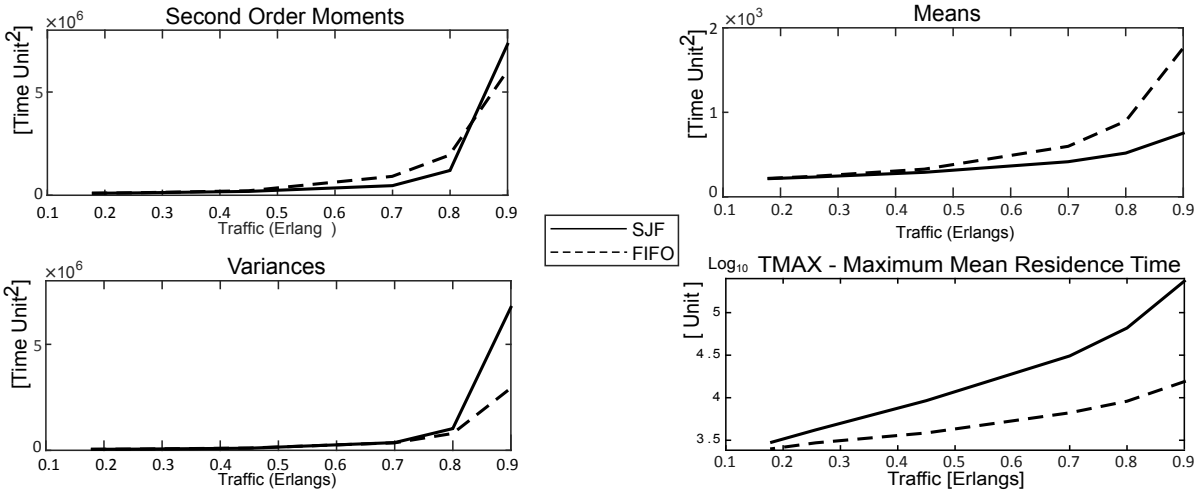


Figura 29 – Moments of the residence times.

Figure 29 shows that in high traffic the second moments is specially important to variance and predictability. Whereas the TAVG^{SJF} is constantly smaller than it is in FIFO (Appendix SJF) , during high traffic (mode C) the increasing rate of the SJF's second moment (its curve's tangent angle) turns its variance sharply higher. It is a predictability reduction

factor. Figure 29-TMAX (note it is a logarithmic axis) shows that TMAX^{SJF} is always larger than TMAX^{FIFO}. Figure 29-TMAX also shows that TMAX behaves quite similar to the variance, and differently from the mean time (Figure 29-Mean).

4.4.3 Results for the CSM

This section deals with the specific issues about mean residence time, its variance both as functions of increasing traffic and their relation to the predictability. Fig. 30 shows the mean residence time ($E[T(x)]^{SJF}$) for different x values.

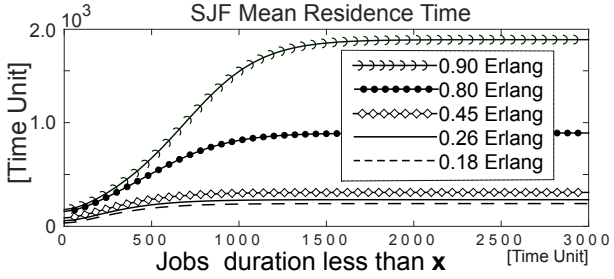


Figura 30 – SJF mean residence time as a function of traffic

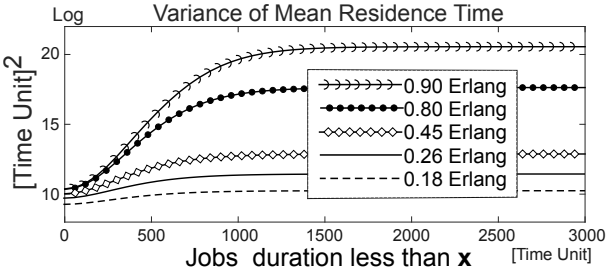


Figura 31 – SJF variance of mean residence time as a function of traffic

The stable (plateau) values are different from each other as the traffic load increases. Whereas, e.g. the load A = 0.9 E, all disciplines have the residence time of 1600 s, except SJF. The system reaches the maximum (stable value) mean value around $x = 1500$ s. Fig. 31 shows the evolution of SJF variance as a function of the value of x for different traffic loads. It is also verified that the values of x that stabilize the variance are different for each traffic load.

Fig. 32 compares the predictability of FIFO and SJF on the x values axis. CSM shows that for small jobs, the FIFO discipline is less predictable than SJF. For the larger ones, it is the opposite. We use one example from Fig. 32.

Up to a certain traffic, the predictability of SJF does not exceed the one for FIFO (e.g. 400 s at 0.800 E). From this perspective, SJF's underperforms FIFO's and tends to enter an unpredictability range of job sizes, and it later returns to the predictability range as seen in the *Analytic Model* from this paper *New Framework*. Up to 400 s in the example of 0.800 E, Fig. 32 shows the values of $\frac{Var[T(x)]}{x}$ of FIFO higher than those of SJF's (in relation to the predictability criterion). This means that for small x values, the FIFO discipline is less predictable than the SJF (a small job can take a long time in FIFO queue). It also shows the opposite for large x values in SJF, especially for high traffic. However, this analysis was

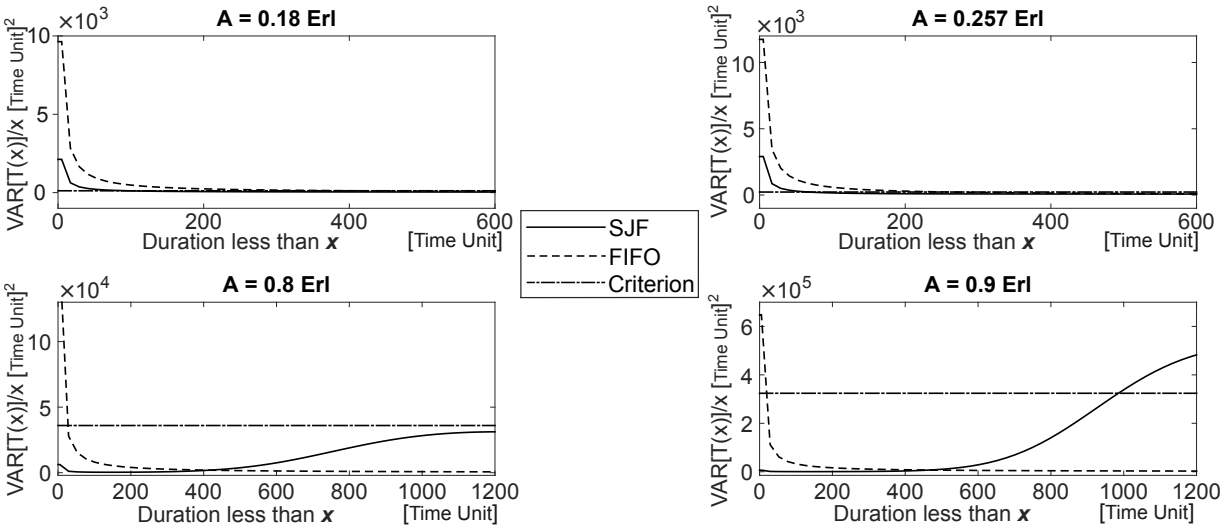


Figura 32 – Predictability of disciplines SJF and FIFO.

performed by CSM focused on specific jobs' sizes, not on a broader overall queuing system performance. Thus, in order to to deepen the analysis, it is essential to consider the approach shown in the next section.

4.4.4 Overall Results

Whereas in the two preceding sections we looked at the results for the DES and CSM individually, in this section we analyze the results from a global and broader perspective, i.e. combining the results from both, as shown in Tables 5 and 6. These tables gather the values of x (from CSM) and TAVG (from DES) and they allow the analysis of the trade-offs between these two.

Tabela 5 – SJF responses for increasing traffic. $TMAX$ from $TAVG_{DES}^{SJF}$ requirement.

Traffic (A)	$TAVG_{DES}^{SJF}$	$E[T(x)]^{SJF}$	x	$\frac{Var[T(x)]}{x}$	$\frac{\lambda E[H^2]}{(1-\rho)^3}$	$TMAX_{DES}^{SJF}$
Erlangs	second	second	second	second	second	second
0.050	[189.34 - 190.52]	189.40	2000	2.04	20.99	727.16
0.180	[215.58 - 216.76]	216.00	1065	25.49	117.53	1119.5
0.257	[233.61 - 234.59]	234.10	900	61.73	225.56	1259.8
0.450	[289.68 - 290.90]	290.30	644	334.47	973.70	2141.2
0.700	[415.02 - 417.80]	416.30	513	1771.15	9334.16	5100.0
0.750	[458.11 - 462.09]	460.10	508	2598.43	17280.00	6340.5
0.800	[516.35 - 521.41]	518.00	510	4088.24	36000.00	8388.7
0.900	[747.42 - 760.44]	755.20	558	18602.15	324000.00	16248.0
0.910	[790.53 - 807.43]	800.30	568	23697.18	449539.21	18045.0
0.920	[2019.28 - 2477.92]	2243.00	1900	1101578.94	647130.23	18936.0

The framework is parametrized to return one x value. Table 5 shows us that the increase in traffic intensity distributes the x values (necessary to produce values of $E[T(x)]^{SJF}$ in the neighborhood of $TAVG_{DES}^{SJF}$) in a regular shape of a tradeoff-curve (bow-shaped) with its minimum point (0.750, 508), as can be seen in Fig.34.

Tabela 6 – $FIFO$ responses for increasing traffic. $TMAX$ from $TAVG_{DES}^{FIFO}$ requirement.

$Traffic(A)$	$E[T(x)]^{FIFO}$	x	$\frac{Var[TAVG^{FIFO}]}{x}$	$\frac{\lambda E[H^2]}{(1-\rho)^3}$	$TMAX_{DES}^{FIFO}$
Erlang	second	second	second	second	second
0.050	189.50	2000	17.95	20.99	727.16
0.180	219.51	1065	45.25	117.53	1075.0
0.257	242.31	900	65.21	225.56	1144.9
0.450	327.27	644	166.30	973.70	1532.4
0.700	600.00	513	701.75	9334.16	2285.2
0.750	720.00	508	1020.47	17280.00	2337.8
0.800	900.00	510	1588.24	36000.00	3177.2
0.900	1800.00	558	5806.45	324000.00	4467.8
0.910	2000.00	568	7044.01	449539.21	4616.8
0.920	2250.00	1900	2665.26	647130.23	4769.3

DES with CSM show that: (1) the SJF policy submitted to an increasing traffic intensity can drive predictable response times to very high traffic values; (2) it can guide decision-makers to reach usually unknown results (e.g., undesirable expected TMAX); (3) it may be feasible to manage the size of a job in the system to meet systems requirements; and (4) it is also possible to evaluate the systems requirements by analyzing the resulting predictability of the size of a job. CSM evaluates the system's performance for jobs from 0 to a given x value and, in another way, DES considers jobs with all the possible jobs to result system average values as in *FIFO* for example (Tab. 6).

We wish to compare both results, so it is necessary to state a criterion. We start out by obtaining $TAVG_{DES}^{SJF}$ (DES mean residence times) to use it as a CSM parameter (Table 5). This implies the use of CSM to obtain the x that results its residence time ($E[T(x)]^{SJF}$ in Table 5 or $E[T(x)]^{FIFO}$ in Table 6) in the neighborhood of the chosen value for $TAVG_{DES}^{SJF}$.

Figure 33 shows the residence time variance for both DES and CSM submitted to SJF and FIFO, through the traffic load (up to 0.90 E). Note that from a certain traffic load, SJF has the highest variance. Additionally, the x value variance for SJF, FIFO through CSM, generated from TAVG_{DES}^{SJF}, is higher than DES values (SJF_{disc}, FIFO_{disc}). In fact, although the first moments have been adjusted, the second moments (in CSM) should not approximate the average values of all jobs. This shows the asymmetry of the distribution of residence time in the system. This result is in agreement with Figure 32, which shows an increase

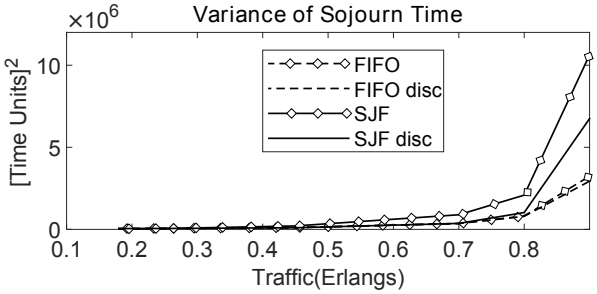


Figura 33 – Variances divert from DES to CSM.

in unpredictability for a range of high traffic values. As x increases, the higher the service distribution H diverts from the exponential. It is sharply noted in the SJF, because it strongly changes (organizes) the exponential distribution within a priority rule, differently from FIFO that does it with minor changes. Figure 28 illustrates that whereas FIFO's service variance remains constant (exponential distribution), SJF's service variance grows significantly. This reasoning is detailed in Section 4.4.5 (Figs 33, 34 and 36).

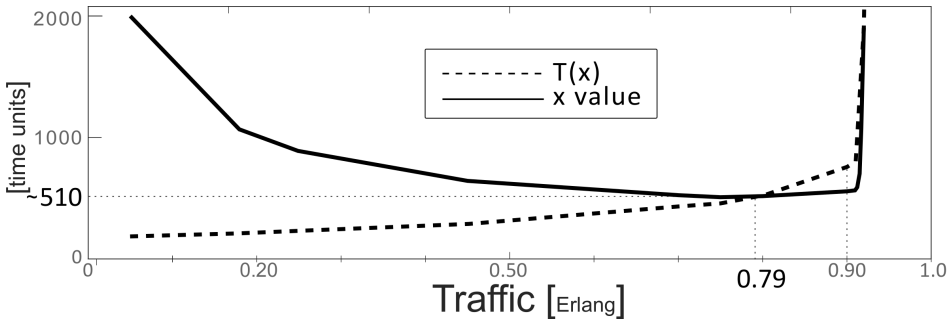


Figura 34 – SJF Trade-off as a traffic function − the x job-sizes versus the framework response for system residence time of the x job-sizes.

Jobs with size x must wait for smaller jobs before they can be served in a SJF discipline. This results their $T(x)^{SJF}$ in the interval ($\text{TAVG}_{DES}^{SJF} \pm \Delta T$). As an example, Fig. 34 shows its minimum x value in 508 time units with 0.750 E (Table 5) near the limit of its increasing traffic. The bow-shaped curve (continuous line in Fig. 34) clearly reflects the most predictable range of x values around 500 time units as a function of the traffic. It should be noted that its minimum point is not enough for better decision-making. It should be taken into account that around this value there are two possible traffic values for each job size (one on the right and one on the left of the minimum). Two other fundamental parameters (maximum time and predictability) should be considered as follows.

Figure 36 shows the logarithm of the variance. It shows that for low traffic (mode A), SJF shows that the variance is worse than the others. For traffic around 0.7 E, SJF is

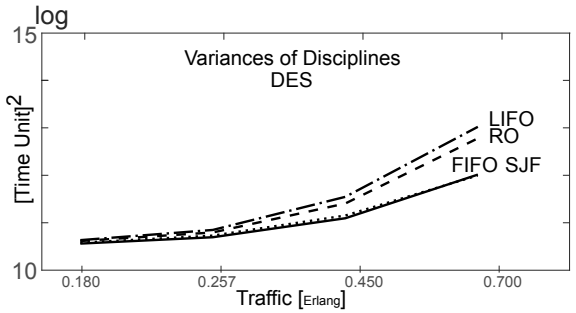


Figura 35 – System mean variance from the DES.

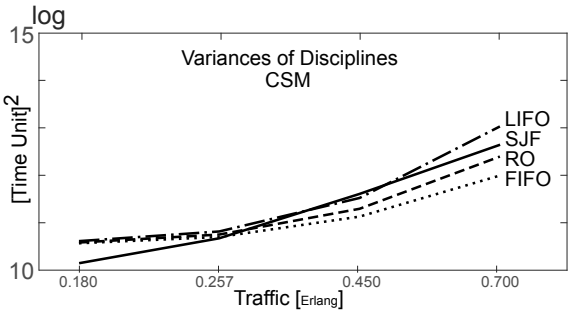


Figura 36 – System mean variance from the CSM.

surpassed only by LIFO. The performance presented in Fig. 36 depicts the variance values considering $T(x)$. It is weighted by the values within the range $(0...x]$ through a given policy and it explains why LIFO is worse than SJF - SJF always selects the smallest job in the queue. Differently, Fig.35 illustrates the same situation just described, but through DES which returns the system mean time. The SJF variance exceeds only the variance for the FIFO policy when subjected to traffic around 0.7 E. The largest x values are hidden within the average.

4.4.5 Discussion

We addressed the framework characteristics, description and applied experiments that illustrate its results. This section addresses a number of issues to discuss the model applicability and generality.

Applicability. The framework applies to single server systems when the job size is known, or when it is possible to estimate them. Given a reference value (e.g. the mean), the framework allows the identification of a range of values of job size that lead the system to under-performance (e.g. a large system residence time above the set QoS value), across increasing traffic conditions. In essence, it allows the analyst to identify how the size of a job affect the system performance.

Predictability. Within this context, Table 5 shows that the predictability (WIERMAN; HARCHOL, 2005) values $\frac{Var[T(x)]}{x}$ are smaller than the $\frac{\lambda E[H^2]}{(1-\rho)^3}$ criterion for all traffic except for *0.920 E*. This means that the comparison of the calculated $E[T(x)]^{SJF}$ values with the reference interval ($TAVG^{SJF}_{DES} \pm \Delta T$) returns predictable x values. Figure 28 shows that the SJF changes the exponential service distribution and this increases its total variance (Fig. 29). As a result, it may decrease the predictability ($Var[T(x)]/x$) due to both the increase

in the job's duration and also in the traffic intensity (Fig. 32). ((WIERMAN; HARCHOL, 2005)) show that, usually, after this increase, the predictability returns to a high level as it reaches the largest job's size (Figure 31). Note that the variation becomes stable at the top of the curve, but the size of jobs continue to increase and $Var[T(x)]/x$ decreases as a consequence. Regarding the comparison of the policies, in high traffic, the predictability of SJF is low, but still better than the one exhibited by LIFO (Fig. 36). Thus, if predictability is achieved under the SJF discipline, this means that RO and FIFO are also predictable. This means that when the framework encounters a value of job size x that limits a given SJF performance requirement in high traffic, this limit may also be applied for analyzing (and possible improving) the performance of other disciplines as well.

Traffic Flow. The framework identifies the size of a job that reduces system performance. This size is usually hidden in the mean value and can be obtained through analysis, e.g. through Fig. 34. This figure shows jobs with sizes from around 500 to about 2000 time units. The trade-off curve represented in Fig. 34 shows the mean time dependency on the x values, specially submitted to traffic over *0.79 E*. The x value curve represents the size of jobs that have a residence time within the chosen interval of the system residence time. The traffic over *0.79 E* and up to *0.90 E* indicates the minimum job size in a system with up to 750 *time units* of residence time. It implies up to 50% of its size in waiting time. The flow of jobs in the system may be incremented within the limit defined by the minimum-sized jobs. Beyond this limit, the jobs with larger size impact the system even more negatively. Thus, this is a flow approach and the solution depends on the adopted strategy. A solution could be taken considering that the SJF's flow is always the best, but considering that the TMAX^{SJF} is always the worst (Figure 29). The mean residence time is always shorter for SJF (Appendix SJF mean). The x value curve in Fig. 34 has a bow-shaped type. This figure depicts that the minimum values represent the large predictable values. Around these 500 *time units* values, it is possible to reach a trade-off value between TMAX and TAVG with an applicable solution.

Decision making. Fig. 34 allows the use of the value of x that is below the predictability criteria (considering the established requirements); it is possible to choose the parameter of the underlined strategy. For example, divert the traffic represented by x-sized jobs to another server, or divide these jobs into jobs in other jobs up to x, or else, change the service policy, as shown in Table 6.

As expected, it is known that, among the non preemptive disciplines, SJF and FIFO are the ones that can better meet the requirements of general single-server queuing systems. However, the framework can use SJF as a reference to compare performance against other disciplines.

The experiments offer Figures 33, 34 and 36 to better manage SJF and FIFO through the framework. SJF strongly changes (organize the sizes) the exponential distribution within a priority rule, differently from FIFO that makes minor changes. It can be observed that the service distribution diverts from FIFO with the increasing traffic (Fig. 28), so does the total variance (Fig. 29 - Variances) and TMAX (Fig. 29 - TMAX). It is deeply observed when one considers the divergence increase due to the job size (Fig. 32). The higher the traffic, the larger the range of jobs' sizes out of the predictability criterion, so, the higher the total variance. It must be carefully managed to ensure flow improvements.

Figure 30 shows that $\mathrm{E[T(x)]}^{SJF}$ takes a longer rising range (intermediate x values up to 1500 *time units*) to stabilize in the presence of high traffic. These intermediate values should be exploited to improve system performance, since that all non-preemptive disciplines (except for SJF) have the same TAVG. Therefore, it is possible to realize that, from a given job size, the system under-performs. It should be interesting to control it (e.g., for 0.900 E with job duration from and above 750 time units) looking for flow improvements.

Considering the issues of traffic volume, predictability, flow and maximum residence time, there are some general results that should be considered to improve system performance. For example, the choice of one or another discipline, or the individualized management (job of a certain size) for any of the disciplines (Table 5 and Table 6 for SJF and FIFO). During system start up, or during low traffic, the policy choice should be the SJF discipline, since it turn the services more predictable. SJF better performs jobs with major durations (Fig. 32), improves the flow rate (Fig. 29) and it do not cause high $\mathrm{TMAX}_{SJF}^{DES}$ (Fig. 29-TMAX). FIFO discipline should be the priority for changing disciplines in a system with very high $TMAX$ or, for very high x values. If this choice may not be taken, RO is the choice. This will lead to better predictability, lower $TMAX$ and better preserves the flow performance. For example, if a system requirement is to reduce TMAX by 50%, it is only the DES that can evaluate the SJF's expected maximum residence time (the last column in TMAX Tab. 5). The mean average time of the SJF feeds the CSM, which in turn evaluates the size of job x that would limit TMAX (i.e. the predictability). For example, considering a traffic of *0.80 E*, the $\mathrm{TMAX}_{DES}^{SJF}$ would be initially reduced from *8000 time units* to $\mathrm{TMAX}_{DES}^{FIFO}$ of about *3000 time units*. The framework may be run for a second time, this time excluding all jobs larger than x, and the resulting analysis may lead to a fine-tuned analysis. This analysis allows the framework to be used to support a planning and management tool.

4.5 Summary and Conclusion

Although research on scheduling and its performance analysis has been around for many years, it recently made an increasing contribution to practical problems through advances in flexibility, optimization, and scalability, which paved the way of a range of scheduling techniques into real-world applications. The gains in flexibility allowed current techniques to be capable of generating schedules under broad and diverse sets of temporal and resource capacity constraints. Regarding optimization, the increasing integration of mathematical programming tools with AI-based search techniques is permitting significantly powerful optimization capabilities. Current scheduling techniques are much more scalable since they are capable of solving large problems (i.e. hundreds of resources and tens of thousands of activities) in reasonable timeframes.

Nevertheless, the creation of high-quality solutions for actual scheduling problems and their related performance analysis remains an artistic craft which is still confronted by issues of scale and complexity and the need for more realistic assumptions and requirements, such as *coping with dynamic changes*. One example of a key requirement is the ability to analyze the performance of a dynamic system where *the traffic intensity is variable*. Above all, current scheduling techniques and performance analysis frameworks are best suited for highly predictable and stable applications (i.e. single-mode applications), where optimized schedules can be statically (i.e. in advance) computed and have a reasonable chance of being feasible. These techniques may have a quite limited lifespan as they were not designed for changing environments which consist of multiple modes of operation and that are highly uncertain and dynamic. Nevertheless, the majority of real-world applications tend to fall toward the other end of the scale, which demands a more reactive system running one or more ongoing processes that respond to unexpected and evolving scenarios. The performance analysis for such systems is lacking in the literature.

Within this context, in this work we proposed a model, a methodology and an implementation of both that allows such type of performance analysis. Through this framework, we showed that under different traffic intensities, it is possible to identify the size of a job that allows the system to satisfy a given QoS requirement (e.g. maximum mean time). The framework combines both a discrete and a continuous simulation module where the mean value obtained from discrete-event simulation is used as an input parameter to the continuous simulation model. The size of the job is an essential parameter that significantly impacts on the performance of the system. Therefore, this work also contributed by showing that it is possible to complement and extend a continuous simulation model with a discrete event model when the CSM model has reached its limitations in terms of complexity. Likewise, the

reverse is also true, i.e. when the DES model is limited in the analysis of certain parameters (e.g. size of a job), the only means to overcome this limitation is to extend this model with a continuous model, such as shown in this work. In essence, we argue that is only through the synergy provided by the sequential operation of both models that is possible to obtain the type of analysis offered in this work.

The practical outcome provided by the framework is that, once the size of job x is identified for a certain traffic intensity, it is now possible to maintain the predictability of the system, i.e. the target performance indicators (QoS requirements) can be kept within bounds. Once the traffic intensity is changed beyond a threshold, a new value of x may be found that ensures the continuity of predictability. This may be enforced by the scheduler by redirecting incoming jobs (with size larger than x) to alternate servers at run time, or by temporarily blocking them, or even splitting the jobs into smaller jobs (offline) whenever conditions allow.

As future work, we would like to address the study of other probability distributions to describe the service in the M/G/1 model, such as Pareto and Weibull. Moreover, for a dynamic approach, we suggest the introduction of a predictor to anticipate critical events in the system such as substantial increases in the maximum residence time in the system (which could trigger a change in the mode of operation to improve overall robustness/responsiveness), thus turning the system more proactive and less reactive. We also contemplate the introduction of a fuzzy-based model to better arbitrate a change from one scheduling discipline to another. We argue that other dynamic conditions (i.e. other than traffic intensity) may also be analyzed and explored by the proposed framework. Thus, another possible research avenue would be the identification of additional dynamic variables or conditions in a system for which we wish to ensure some sort of predictability (e.g. either job, variable or policy predictability) resulting from the application of the framework.

5 Simulation Model for Performance Analysis of a Hybrid-Policy (SJF/FIFO) Parallel-System

This work deals with a system with multiple servers in parallel in which the duration of the service in each of them is known (or it is possible to estimate), each one working independently, with non-preemptive discipline. Completion of the service provided presupposes that a part is executed on each server. In Chapter 2 we work with several servers in parallel, each working independently of the others, and each arrival request to be completely executed must pass through all the servers. Moreover in Chapter 2 2 we introduced the DES model as a black box, and we discussed its functionality as a framework of a hybrid simulation model discrete and continuous (DES/CSM) that evaluates a full SJF scheduling model for a single server queue and their behavior towards non-preemptive disciplines. In this chapter, we look into the DES model as a hybrid policy SJF/FIFO model for multiple parallel servers, i.e. we expand this general view by examining each of its constituents in more detail and how they are related to each other.

5.1 Introduction.

Real-world changes and instabilities (e.g. unpredictability of input traffic) are often partially known, or predictable in some specific circumstances (e.g. subject to data history or a few analytic models and specific algorithms). Thus, the unpredictability that often happens dissociates the real-world systems from the models that represent them (BANKS *et al.*, 2010b; DEVORE, 2004; HARRELL *et al.*, 2012). This is the case of common applications in supermarkets, call centers, public services and production lines to name just a few. In this way, the need for performance analyses that involve issues such as unpredictable changes can be considered as an actual requirement. As the system's conditions (internal or external) change, a new analysis is required to evaluate the new possible constraints (MARTINS; BURNS, 2008). Whereas these applications may involve a number of servers, for the sake of simplification without compromising the results, this work adopts a system of three independent parallel servers (Fig. 37). This system falls into the case of a network problem of generalized queues containing in its core a closed system (BITRAN; MORABITO, 1996). Each arrival entity must pass through the three servers before service is completed (a *com-*

pleted entity) and so, leaves the system, otherwise it is an *uncompleted entity* (Fig. 37) and return to servers that still need to attend it. More details about this approach can be seen in the Santos *et al.* (2018) previous work.

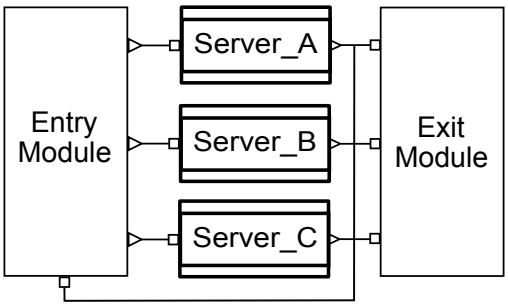


Figura 37 – Representation of the three-server model ((SANTOS *et al.*, 2018)).

The new model becomes more realistic when implemented with changes in the traffic intensity, and its resources and processes closely map to the ones from the physical world. The key element of the proposed model is a hybrid and dynamic meta-scheduler which is able to select at any time the scheduling policy that meets the systems requirements at that particular point in time. It is possible through the proposed model to choose the percentage of scheduling policies for the system (e.g. 30 % SJF and 70 % FIFO). Clearly, this implies that allows that some inbound jobs are queued in order of arrival (FIFO) whereas the remaining are queued according to their service time (SJF).

One of the underlying motivation of this work is due to the progress achieved lately in modern queuing systems, e.g. waiting rooms in bank branches, medical care etc . In this systems, the customer is required to fill up an online system which prompts for the type of customer (e.g. elderly, VIP, special care, conventional), the type of operation (e.g. deposit, transfer, exchange). Other types of information may also be requested depending on the application. From a technical perspective, this is an indicator of job sizes and priorities for each individual, which may be grouped into a specific class. The scheduling policy then factors this information to make the selection of the next customer(s) to be served. Nevertheless, current policies in place are still empirical, in the sense that, whereas they attempt to satisfy the requirement of fairness, the requirement of high performance is not effectively considered. Therefore, a new approach to scheduling that accounts for a further analysis of the jobs' classes that promotes performance and fairness needs to be developed to exploit further gains in system performance. In this scenario, certain classes of customers reduce system performance ((MOR, 2010). The identification of these classes in a dynamic application with an intermittent flow of customers in and out of the system, and the solution to this, may translate into substantial gains in performance. Within this context, this paper proposes a model, which

is implemented using DES for the performance analysis of complex systems across dynamic changes (e.g. input traffic). This challenge is exacerbated by complex processes subjected to changes, lack of stability, among other possibilities.

It is intended that the model satisfies two propositions. Firstly, to allow the planning of the system's operation and the allocation of resources ahead of time whenever necessary. Considering the complexity and the changing conditions of the system, this planning may not be easily obtained via conventional models and methods (e.g., branch and bound, Jackson Networks and general equilibrium model just to cite a few). Such planning is usually an offline (i.e. static) activity with a perspective in the future, be it short, medium or long term. Secondly, to allow the management of the system operation in view of the unexpected conditions. Whereas this also involves the allocation of resources within the system, this is often the case of an online activity, usually with a perspective in the present. For example, with the online performance information supported by the planning information, the analyst/manager is able to previously identify possible bottlenecks and make decisions and take actions to avoid unwanted results. More specifically, the main questions that the simulation model intends to answer are: 1) Does this simulation model allows the identification of variables behavior based on the dynamic change of scheduling police SJF/FIFO considering changes in the traffic intensities (Erlang [E])? 2) Can the model be used as a management support tool? 3) Does the model ensure the maintenance of a certain established QoS requirement? For example, consider an application that needs to keep a given TAVG limited by a maximum value to satisfy the system QoS. 4) Can the model estimates a change in the upward trajectory of the TMAX to reduce it to a level below a certain value (i.e. a set point defined by the application)? Can the model keep the system QoS within the established requirements?

The simulation model allows, as previously mentioned, the identification of how the system performs, in terms of TAVG, TMAX and TSTD, through number of entities in SJF/FIFO policy subjected to different traffic. This feature enables one of several strategies to ensure the established QoS requirement. A simple action could be to prevent jobs sizes greater than a chosen job size (e.g. the relatively large jobs to be processed) from entering the servers used at the general service, e.g. by diverting these jobs to an auxiliary (or secondary) server, or by promoting a temporary blocking.

The remainder of this paper is organized as follows: Section 5.2 discusses related work and some other applications; Section 5.3 describes the proposed model, Section 5.4 introduces the implementation of the model as discrete event simulation, Section 5.5 presents a case study that illustrate the application of the proposed approach, In Section 5.6, we analyze the results. Section 5.7 addresses our remarks and discussion, and Section 5.8 presents our

conclusions and suggestions for future work.

5.2 Related Work

In this section we present and discuss some examples of practical processes as shown in Fig. 37 in a variety of scenarios and different types of operation.

An actual example of logistics for the food retail market is the process of loading trucks (entity is the truck, commodities are jobs). The whole cargo carry many commodities types (e.g., corn, bean, rice, etc.) that can be randomly laden in trucks by servers availability (SANTOS, 2014). Xu *et al.* (2014b) show a Medical emergency procedures (entity is the patient, services are jobs, independent or not) that uses $IoT-based$ methods where a chain of simultaneous procedures with responses coming through the Internet can guarantee patients' lives. This specific example shows that the distance between different steps (servers) that make up a complete process is no longer a problem (EECKHOUT, 2016b; FEKI *et al.*, 2013b). Humanitarian aid (people are entities and services are tasks) is another example with a structure in the form of queues and service stations that needs to streamline assistance processes for people with many types of needs. Servers perform simultaneously care for people. As soon as a server ends the service, the patient gets in line for another service and will not leave the system until all his/her needed services were attended (SANTOS *et al.*, 2018).

In general words, the proposed DES-model is the main structure of a mapping-method that identifies how the system's variables interact and allows the performance evaluation of a system with multiple classes of services based on *JN* submitted to hybrid *FIFO/SJF* scheduling and varying traffic.

Sandmann (2006) promotes fairness performance evaluation for DES in systems modeled by queues *M/G/1* subject to hybrid policy *SJF/FIFO* alternating one entity in *SJF* and another in *FIFO* (50% of alternation). In contrast, our method adopts *DES* associated with incremental validation for systems with entities formed by n classes of tasks (e.g, different service averages) served by three parallel servers subject to the hybrid policy with selective control of *SJF* over *FIFO*.

Kim e Kim (2015b) provides an evaluation method (for medical emergency care) based on DES and a hybrid scheduling of entities classes - prioritized class and FIFO class. Unlike, our mapping and evaluation method based on DES with incremental validation (first) applies for systems with multi-task entities (and second) with selective control of hybrid $SJF/FIFO$ entities, i.e. the method uses a DES model that controls a specific number of SJF entities into the hybrid cycle to plan and manage the system performance.

An evaluation method based on DES is studied by (CRUZ; DADUNA, 2017b) for optimal inventory allocation of multi-class entities in single server systems. Differently, our work is focused in a mapping and evaluation method based on DES with incremental validation applied for multi-task entities attended by systems of servers in parallel topology.

A hybrid method based on policy reorder plus numerical method plus DES is proposed by (HUM *et al.*, 2018) to evaluate responsiveness of supply chains in specific multi-stages and parallel topology considering possible backwards transitions. Authors consider responsiveness in terms of the probability of fulfilling customer orders within a quoted lead time. Otherwise, our method based on DES and incremental validation estimates a point to point performance of a set of requested variables for two-stages systems of n parallel and dedicated servers with re-entrant multi-tasks entities.

The revision work of (BRANKE *et al.*, 2016) brings the almost optimal solution method to Hyper-heuristics, each specialized and applied to a particular type of problem. Unlike a specialized methodology, our method is generalist in promoting the mapping of the behavior of variables; and in the opposite sense of the application of exact solutions, it contemplates unpredictability and complexity when using DES associated with incremental validation to approach the real-world system to the maximum similarity.

Venkatapathy *et al.* (2017b) propose the use of the dynamics of intra-logistics (associated to cyber-physical systems) with a system dynamics approach for optimizing complexes systems in terms of logistics availability of information, objects and persons. Our focus is the use of simulation-based mapping (if also associated to real-time data communication in cyber-physical systems) to provide management functionality as a dynamic support of intra-logistics during emergent deviations.

5.3 The Proposed Model

5.3.1 Preliminary Considerations

The decision for the JQN analytic model was made because the M/M/1 model with parallel queues with entities passing through more than one server fits for the real-world application (SANTOS *et al.*, 2018) this work approaches. The SJF scheduling as a second policy was chosen because 1) the central proposition of this work is to trade-off the values of TAVG and TMAX against the FIFO's values; 2) SJF is intuitive and widely used in real-word applications; and 3) it is widely recognized as the non-preemptive scheduling policy that best optimizes the throughput of non-preemptive systems as seen in Chapter 4 and in Mor (2010) and also in Santos (2014). The classifying process is necessary to enable characteristics'

analyses of the outbound-variables' classes (TMAX's tail is an example). SJF and FIFO policies have to be independently validated with the model that toggle them to trade-off TAVG and TMAX. Considering a planning perspective, the model is intended to identify the system's behavior (variables sensitivity and their relations to each other) by estimating some variables' results, submitted to the conditions later proposed. Regarding the managing perspective, the model also allows for keeping the system within the state requirements. Firstly in a general sense, with the availability of the estimated planning information, to previously identify bottlenecks, and secondly, to take the necessary managing decision to avoiding the identified bottleneck. Also in the context of the managing perspective, the discrete-event simulation model can perform the hybrid policy with actual data to complete the system's runtime (Fig. 38) to support short time estimation of the system performance.

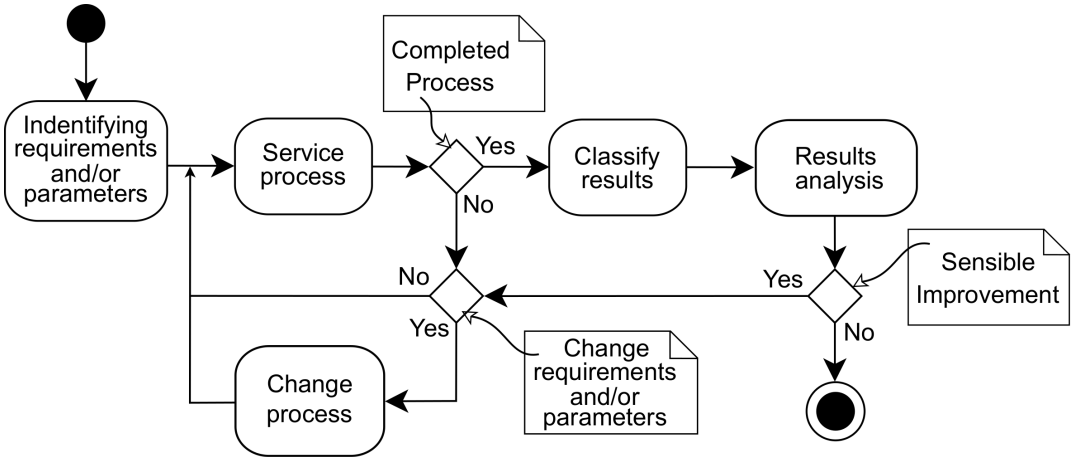


Figura 38 – Activity diagram for the hybrid-policy proposed model

Each cycle time is represented by one full DES steady-state runtime that begins with the SJF policy and changes (as a setup value) to FIFO when the total number of entities in system reaches a given number N. Considering that each runtime is executed by an inflow period of entities (going to a top value) followed by a period of entities' outflow until a bottom value, the system may have two different N numbers: N_{in} for the inflow and N_{out} for the outflow (see Fig. 47 in Section 5.6). In other words, the system may have an inflow setup and an outflow setup. Pre-defined system's requirements or QoS requirements establish the values of N as a kind of cutoff values, named *the system's steps*.

With this design, the model allows the estimation of the system's performance subjected to progressive effects of a control's parameter that, in this case, is the number of jobs that enters the system during the SJF policy. For example, different traffic intensities may cause system's responses that either fit the stated requirements or not. A sequence of results from a system's variable (TMAX is one example) that is estimated from simulation during

the offline planning process, or measured during an online real-world operation, is considered to be a *system's path of results*. Different traffic intensities may be represented by different results' paths (trajectories) in a *performance map* (as in the surface of responses graph of Fig. 53) of this hybrid system. Having the performance map previously in hand, the analyst can understand the system's limits and manage a set of performance strategies to facilitate the avoidance of bottlenecks and undesirable results.

The classifying (Section 5.4.4) function of the DES model groups the outbound jobs into different TAVG's classes in a first result level. The next level evaluates which job sizes x compose and impact each class. This information enables the analyst to manage the system's responses on a job-size basis. In this way, the manager adds control to the entities whose job sizes have a negative impact on the required results. This model allows the analyst to manage the operation through the results' path of the observed variables and adjust the course of the results to another path that keeps the system within the established requirements.

The case study (Section 5.5) explores the shortest mean maximum residence time (TMAX) of FIFO associated with the lowest mean residence time (TAVG) of SJF. In this way, this work represents a small advance in the approach of performance analysis of the scheduling of jobs subject to unpredictable changes of the incoming traffic intensity in systems that seek to keep a QoS requirement under control.

5.3.2 Model description

The model consists of requests, jobs and resources. This is a real-world logistics-application of charging products into trucks and, requests are named *customers' orders*, jobs are the *products* and, the resources are the *servers*. This work defines requests as entities and an entity, as a set of three jobs. Each job is a task that have to be processed in a server (a resource) during a period of time named as the service time. This period is also understood as the job's size. The arriving time of entities and the service times are random variables to represent a queuing model, i.e., this is a system with inflow and outflow of entities that trade-off for servers. This characteristic allows the use of the TAVG/TMAX trade-off of the SJF and FIFO policies identified in Chapters 3 and 4 to build a model that looks for improvements on the system performance.

The key element of the proposed model is the dynamic meta-scheduling (i.e. hybrid approach) of two policies , e.g. SJF and FIFO, as described in Fig. 39. The model schedules an entity to a server or by the order of arriving time or, by the order of job's size. The policy change logic is detailed through two main activities: the *Classify Flow* and the *Toggle Policy*. The policy change activity compares the number of inbound entities in the system with the

setup value of a variable named varTHRESHOLD. The comparison result drives the change on the current scheduling policy (or not). This logic continues until either the number of jobs in system is smaller than the threshold, or the system is empty (all the available jobs have been processed), or also when the system is subjected to overflow.

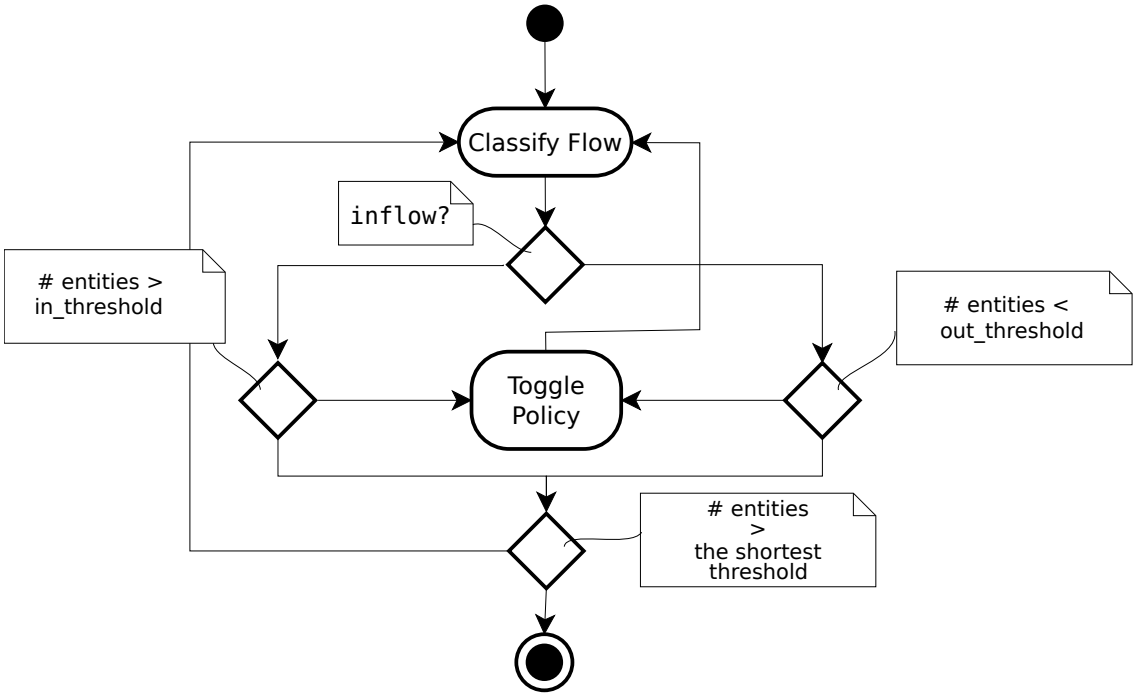


Figura 39 – Activity diagram of the policy-change algorithm.

A control module named *Hybrid Policy Control* (described in Section 5.4) effectively provides the policy changes. Each algorithm (SJF or FIFO) searches the requested buffer for the first available job according to the scheduling priority at the moment of the search what ensure no residual jobs from the earlier policy.

The hybrid control functionality was incremented and validated in two phases: the *Policy* module and the *observer* routine. The *Policy* module, which integrates the *Hybrid Policy Control* subsystem, sets the policy change point by comparing the number of entities in the system with the variable $varTHRESHOLD$ that defines the threshold point N of the policy changes. A pair of variables ($varUP$ and $varDN$) informs the system whether the number of entities is increasing or decreasing, in other words, whether the system is in the process of *inflow*, or *outflow* of entities.

Consider Fig. 40 that represents a system starting out with the SJF policy (left side of the upper graph), which shows in the lower graph the total number of entities in the system in a timeline. The number of entities is a random variable defined by λ. The large-dashed and the dashed lines represent (respectively) inflow (i.e. rising) and the outflow (i.e. falling)

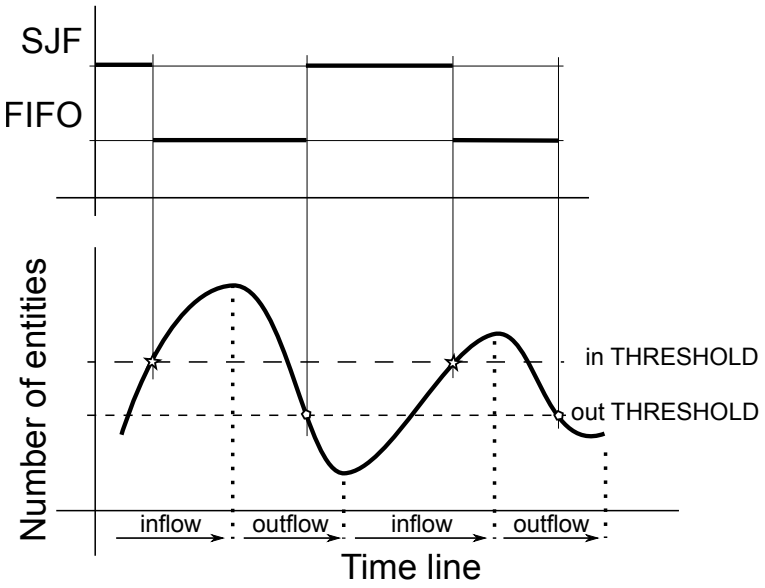


Figura 40 – Policies sync with inflow/outflow cycle.

thresholds. The scheduling policy is toggled under two scenarios: 1) The inflow crosses the in_threshold line (large-dashed line), or likewise, if 2) the outflow crosses the out_threshold line (dashed line). Note that, if only one threshold is defined, the scheduling policy is toggled once the number of entities cross the threshold line. It follows then, from the definition of a threshold, that for a system that is mostly moderately loaded (50-70 %) and has a relatively low threshold, it means that the system will run mostly a FIFO policy (assuming that it started out as SJF) due to there will be few queues. A system that is mostly lightly loaded and has a sufficiently high threshold also implies the predominance of FIFO over SJF. A system under high load and with a sufficiently small threshold also implies in the SJF policy running most of the time. Finally, it is clear that for a system that has a threshold set at a value that is the mean of the total traffic, the changing of policies tends to be much more frequent if the traffic oscillates around the mean. A threshold that is large in relation to the absolute number of entities at any given time implies that the system works mostly with the initially chosen policy (SJF in this case). On the other hand, a threshold that is small in relation to the absolute number of entities at any given time implies that the system works mostly under the policy that is not the one the system initially ran (the SJF policy). Differently, when the system is under traffic overflow it is expected that the number of entities will go up indefinitely as seen in the overflow period in Fig. 41.

This is a consequence of the low capacity of processing jobs considering the entry's traffic ($\rho = \lambda/\mu \geq 1$). Some expected facts are known to reduce or to stop the overflow period. One possible fact is a spontaneous reduction in the traffic intensity. A second one is to stop entities from entering the system, and the other one is to divert traffic to auxiliary servers.

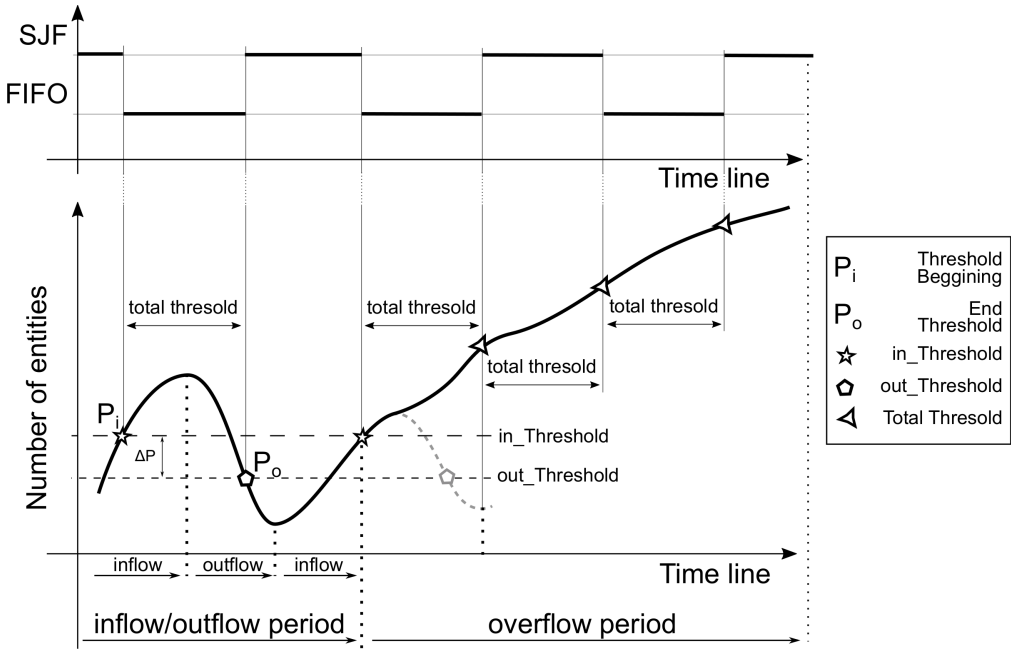


Figura 41 – Schema of the Sync signal responses for the dynamic toggling of scheduling policies. *Upper graph* - A graph of the changing policies sync by the total number of entities in system. *Lower graph* - It represents a double-threshold system where the total number of entities triggers the policies' change according to the in_Threshold (inflow), to the out_Threshold (outflow) and during the overflow period (total threshold).

During overflow, or any increasing period larger or equal to the *Total Threshold* value, the proposed model toggle the policies each time the entities' quantity in system increase in a number equals to the *Total Threshold*. The total threshold is the difference (in the time line) from the values of *out_Threshold* and *in_Threshold*. As a numerical example, admit a system with two threshold values: in_threshold $P_i = 20$ entities, out_threshold $P_o = 16$ entities and $\Delta P = 4$ entities. $\Delta P = P_i - P_o$ is defined as shown in the lower graph of Fig. 41.

The proposed simulation model, despite of the traffic, toggles the policies considering this threshold value established by the system requirements as seen in Fig. 41. In an example of a single threshold system, the overflow's threshold is the threshold itself. Considering a regular operation in a system not in overflow, we now look at why the inflow is used as a condition to toggle the policy beyond the inflow-threshold, and similarly, why the outflow is also used to toggle. Recall that the main goal of the managing function is to keep the performance variables such as TAVG and TMAX within acceptable bounds. As shown in Chapters 2 and 3, this is not easily accomplished with one single policy (e.g. one would have to resort to other strategies such as admission control, which is not our focus).

In short, SJF is known to be the policy that results in the lowest TAVG values. On

the other hand, FIFO results in the lowest TMAX. Thus, in order to keep both TMAX and TAVG within bounds, the proposed model switches (i.e. toggles) between the two policies. Notice that this is dynamically carried out during system operation. When TMAX rises up to too large values (i.e. beyond a certain threshold), the system switches to FIFO in order to "alleviate"/reduce TMAX until it returns to the acceptable bounds. By doing so under FIFO, TAVG is increased over time, which in turn prompts for a switch back to SJF as soon as possible. Thus, the model toggles to the SJF policy to allow the system achieve the shortest TAVG. The cycle is repeated over again once TMAX crosses its threshold or it is kept within acceptable values. Clearly, the value of the thresholds must be obtained from field applications and from the simulation model which is discussed in Section 5.4.

5.3.3 Model Applicability

The model can be applied to both planning and dimensioning the system, and to the system's management restricted to three conditions. The first one is the general and main condition applied to systems which jobs can be previously identified or estimated. Second, in the case of planning, when it is possible to begin the system modeling with a known and validated model, and the third condition is, when the initial model allows the addition of small increments in order to fit the applicable model as close as possible to the necessary real-world characteristics.

Whereas the usual approaches depend on closed mathematical/statistical models to deeply represent a problem, the proposed model is based on DES, dynamically toggling a hybrid scheduler that selects one of the two disciplines (it fits to more than two) to control the number of its admitted jobs with partly SJF policy and partly FIFO policy of a limited capacity system. With this design, the toggling for an eventual decision making on changes of new system conditions, as e.g., scheduling policy, or production arrangements are no longer function of the system run-time, but, as in this work, the number of jobs in system.

This model presented in this section can be implemented in most of the commercial software (e.g., *ProModel*® and *Arena*®, which is the one used in this work) through their *Academic Versions*, as shown in the following section.

5.4 The DES Model Implementation

The model was implemented using DES with several rounds of simulation (simulation's replication) that in each one of them there was a different point of change between the disciplines SJF and FIFO (remembering that each replication started with the SJF dis-

cipline). This toggling point of disciplines reflected the number of active instances in the system for switching from one discipline to another.

The proposed initial simulation model in Santos (2018B) was extended to approximate the real-world model (Fig. 42).

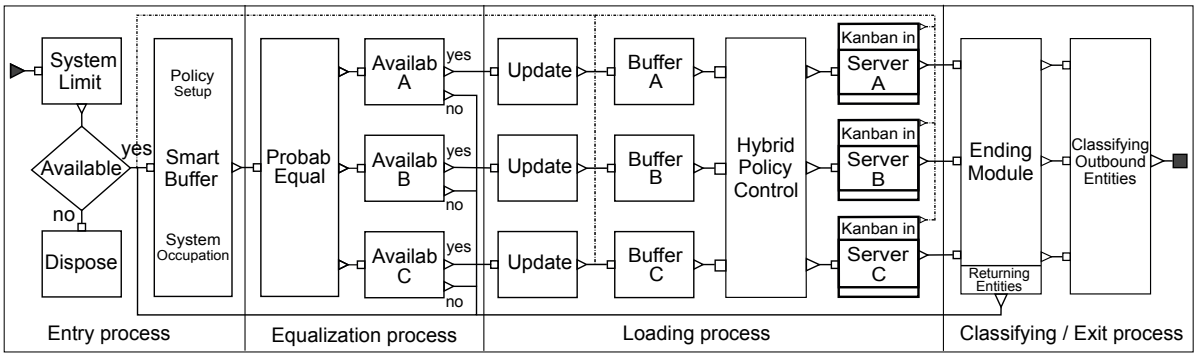


Figura 42 – Hybrid Policy Model. *System Limit* disposes undesirable overflow. *Probability Equalizer* guarantees equi-probabilities to every server. The servers buffers are independent queues.

This design allows the use of increments with other distribution probabilities as exemplified in Section 5.1 to eventually change the implemented exponential distribution. Figure 42 is organized in four logic sections - *Entry process*, *Equalization*, *Loading process* and *Classifying/Exit*.

5.4.1 The Entry Process

The simulation model was developed with an entities generator that capture the system parameters and attributes both time between arrivals (λ rate) and the service times (μ_i). The *Entry process* controls the total number of entities to meet the actual system capacity or software limitation stated in the setup Variable *varINLIMIT*. The *System Limit* process does it by comparing varINLIMIT with the actual number of entities in system (calculated by the difference from the entering entities and the exiting entities) updated through the variable varNUMENT (it can be also understood as the number of work in process entities - WIP). This is the key to allow an entity to enter the system or, to dispose it if the system is overloaded. The *Smart Buffer* identifies the inflow or outflow process to a policy changing as in Fig. 43.

There are two parts in the switching logic of Fig. 43: the inflow/outflow and the overflow periods. The first logic sets the system in a regular cycle of fill-in/emptying entities in a short sequence of time. To meet this functionality described in Fig. 40, the logic compares the number of entities updated in the variable varNUMENT to the threshold value. In a

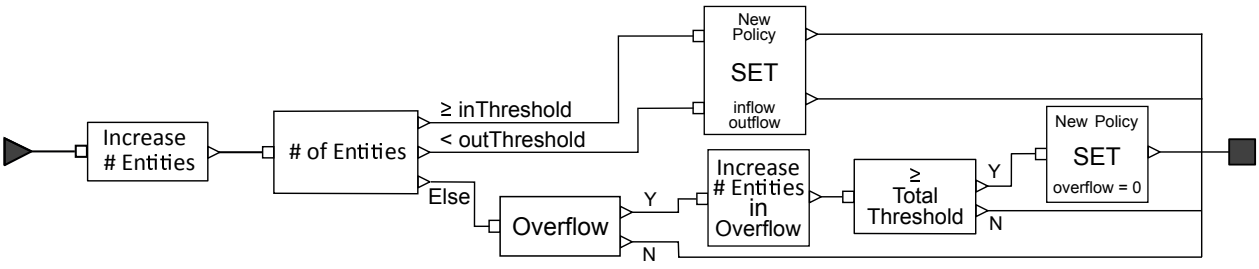


Figura 43 – Policy setting Toggler. The logic to switch policies between SJF and FIFO.

double threshold system as earlier described, if varNUMENT floats from $\geq$ in_THRESHOLD to $<$ out_THRESHOLD, policies changes in a regular basis from SJF to FIFO and vice-versa. Differently, if varNUMENT goes from $\geq$ in_THRESHOLD and can't be back to the out_THRESHOLD, i.e. the system came into *Overflow*, the toggling process switches each *total_THRESHOLD* value increase in the number of entities.

5.4.2 The Equalization Process

The *Equalization* process distributes (SJF or FIFO) entities with equal probability to one of the buffers (Fig. 44) of only one of the servers the entity did not receive service. The *Availab X* blocks (X is the server number) (Fig. 42) set up an availability variable named $varIN\_X$ (X is the server number) to close the access for a next job to the same server and open up the server's buffer access to the requested job, otherwise it is always unavailable.

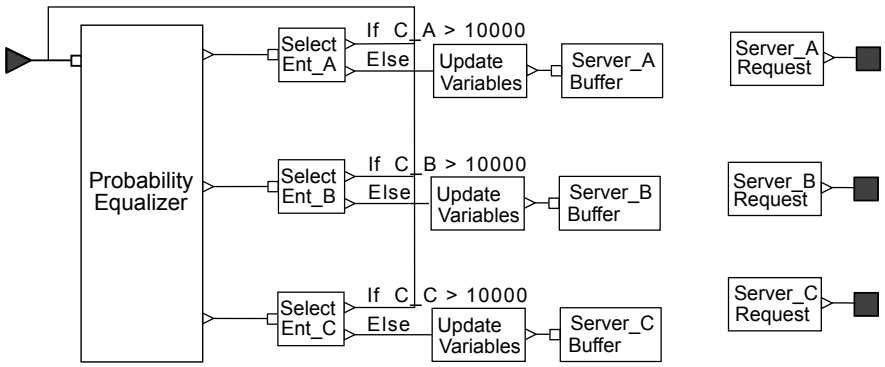


Figura 44 – The Probability Equalizer routine. guarantees equiprobable distribution of entities and feeds the *Observer* routine in a Kanban procedure driven by a server.

The whole system contains blocks *Update* that interchange data from/to entities, resources and the environment. The system exchange information, e.g., the Sever_B is free for the entity, or the entity is unattended by Server_A and Server_C. In a general way, they

represent inbound and outbound actuators for any system information with an automation initiative as, e.g., *RFID/Digital Twin* systems devices (SCHLUSE *et al.*, 2018b; XU, 2012).

5.4.3 The Loading Process

The *Loading Process* actually represents the process of laden products on trucks (in general applications, it is the serving process). Entering the *Loading Process*, an entity waits to be processed in a server's *Buffer* that frees it according to two conditions: the first one is a Kanban sign (OHNO, 2009; LOLLI *et al.*, 2016; PIPLANI; ANG, 2018) sent by the *Kanban module* associated to the server as it is just free; and second, the scheduling policy defined in the block *Policy* by the policy control requirements.

5.4.3.1 Kanban Observer

The *observer* routine represents a Kanban procedure for requesting a new job to be processed on a server. The job is removed from its buffer (part of the process *Probability Equalizer*) according to the policy in force at the time the job leaves its buffer (Fig. 45), similar to a model of pooling represented by Fig. 46. This kanban routine allows an entity to be effectively attended by the *Server* with the correct policy to avoid the presence of entities called in a later policy.

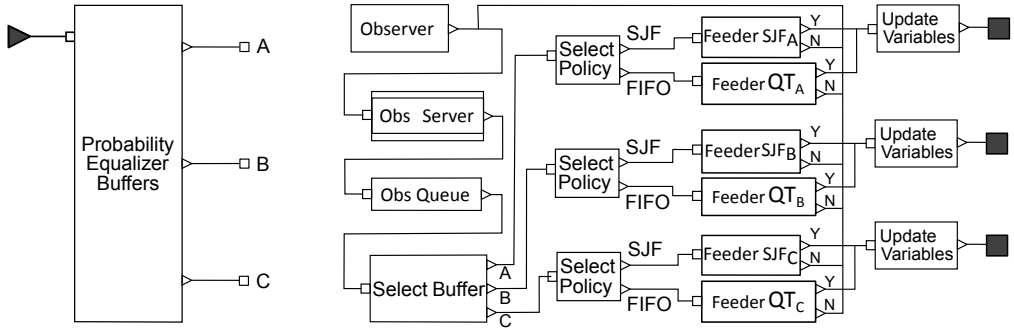


Figura 45 – Distribution Module schema. (a)The *Probability Equalizer* routine guarantees equiprobable distribution of entities. (b)*Observer* routine is a Kanban procedure driven by a server.

5.4.4 Classifying Process

The *Classifying/Exit* process outlines two tasks. A router that returns instances with not-provided services to the *Equalization* process and also, the *Classifying/Exit* logic sends the finished entities to be ranked into values. It allows the necessary data analysis of histograms that can represent output probabilistic distributions (e.g., number of entities per classes of

sojourn times) and the identification of critical sizes of jobs. Entities finally exit system. inflow and release (outflow). Finally, the next section discusses the characteristics of the Mapping Method and presents a practical case.

5.4.5 Other Considerations

We started modelling by a simple and validated JQN modeled with DES. The complex features (that seeks to approach the real-world) were added through InV in small steps (HÄHNLE; MUSCHEVICI, 2016; HAYNES; LENCH, 2003). Thus, the model respond with stationary states independently of the modeling formalism. It also aligns successive estimated responses of a control variable grouped into classes. The set of results is aligned and forms a path (trajectory) of responses. The set of estimated paths is a Map of the system's responses as an example of a graph of surface of responses (Fig. 53). Each trajectory (LEYE *et al.*, 2014) stems from a controlled and selective incremental interference (SJF policy) set as a new characteristic in the initial validated model. This interference is controlled in the sense of the cutoff points that form the paths and it is selective because the system can choose the interference's intensity. In our work, interference is the SJF policy and, therefore, the concept of intensity represents the range of cutoff values in an inflow/outflow cycle during the policy. As an example of the selective interference, the SJF policy may start in the point of 20 entities in the system during the inflow and finishes with 16 entities of the system's outflow. The analysis of the SJF incremented model returned the expected reductions of mean-time values (compared with FIFO). It shows an increase in the number of released entities in the shortest range (0-100 ut). The distribution of the SJF's residence times is an adjustment applied to the FIFO's one. The shorter SJF's mean residence time happens together with a shorter half-width. This fact identifies a higher density of jobs around the mean residence time of the SJF policy, the decrease of the number of entities in the classes with high residence time is compatible with the sense that the farther away a job's class is from the mean residence time, the less entities belongs to the class (a consequence of the Kleinrock's conservation law (**??**)). The extreme high values of TAVG (> 600 ut) are less than 0.5% of the total number of jobs leaving the system. In contrast, the SJF's $TMAX_S$ variable values submitted to the highest traffic values suffer intense increase. These results justify the hybrid model built with two algorithms for non-preemptive scheduling policies - the FIFO and the SJF. A set of requirement variables trigger these algorithms to dynamically toggle between both policies as in Figure 46.

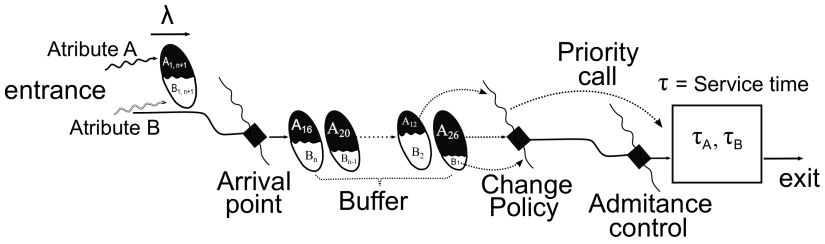


Figura 46 – Change of scheduling policies considering two entities' attributes. Model adapted from(MLINAR; CHEVALIER, 2016).

5.5 Case Studies

Due to this work is an approach of a typical logistic case, it is important to retake (as in earlier Chapters) that the acronym of *time unit* is ***ut*** (Table 7). It is a decision taken in order to avoid the misuse of the symbol *TU* that means *transportation unit* (JANIĆ, 2014). There are two cases in which the behaviors of TMAX, TAVG and TSTD are analyzed as functions of the threshold values. The system in case one is subjected to both the SJF/FIFO hybrid-policy and traffic's instabilities and, in case two, the same system is submitted to traffic overloading. The third case study is a conceptual surface map for managing purposes.

Tabela 7 – Simulation parameters for cases 1-5

Replication parameters		Traffic parameters (ut)	
Number of replications	100 units	Inter-arrival time ($I_t = 1/\lambda$)	21, 20, 19, 18
Replication time	100,000 ut	Service time A ($B_A = 1/\mu_A$)	20
Warm-up time	10,000 ut	Service time B ($B_B = 1/\mu_B$)	15
Confidence level	95 %	Service time B ($B_C = 1/\mu_C$)	10
Total system capacity	145 entities		

Table 7 describes the simulation parameters to all the presented cases as follows: the adopted replication time of 100,000 ut is sufficiently greater ($\approx$ 40 times) than the largest values of mean average time ($\approx$ 2300 ut) of all the cases in this work; the warm-up period was set to 10% of the replication time. The total system capacity of 145 entities respect the limitations of the *Arena*® Student version to represent a real-world operational capacity; the use of the 95% confidence level is because it is broadly accepted in all research fields. The 100 units of replication fits the general statistical needs of repetitions; the model also considers the mean service times for each server to be exponentially distributed because it represents the mean size of the products to be served. Together with the simulation parameters there are also one *measurement parameter* and two *toggling parameters*. The *measurement parameter* is the value to be increased (e.g. each five entities) in the system controlled variable, i.e. an increment to the total number of entities in the system when it is subjected to the

SJF policy. The *measurement parameter* (in the example of five entities to be increased) means the model, e.g. starts out a full simulation to toggle the policies with five entities in system to the completion of the first simulation process. Sequentially, the model repeats this simulation process to toggle the policies with 10 entities in system until its completion and, repeat it again with 15 entities and, again with 20, 25, 30, 35 and, so on until the total system capacity. In short, the *measurement parameter* is the increasing value applied to one controlled variable that allows the observation of system variables (e.g. TAVG and TMAX) through the controlled variable (e.g. the number of entities in system). The *toggling parameters* are the $in\_THRESHOLD$ and the $out\_THRESHOLD$ earlier defined. These parameters are used across the cases summarized in Table 8.

Tabela 8 – Case Studies

Case Study	Description (Goal)	Toggling conditon
1.1	TAVG and TMAX vs # entities	no toggling
1.2	TSTD and TMAX vs # entities	under overload
2.1	TAVG vs # entities	toggling
2.2	TMAX vs # entities	under
2.3	Toggle activity	overload
3.1	Conceptual surface map	toggling under overload

Case 1: *Analysis of TMAX, TAVG and TSTD as a function of threshold values with the system not submitted to overloading.*

This case was run with inbound traffic of 2.5000, 2.3684, 2.2500, and 2.1429 E for the hybrid SJF/FIFO system, but the toggler do no operate during the overload. The model considers a random traffic cycle in a sequence of inflow and outflow periods that both inflow and outflow traffic cross one unique threshold value. The model processes the part from the hybrid policy that corresponds to the SJF scheduling with the total number of entities stated by the threshold that vary each five units in new simulation runnings (as the earlier example) until 145 entities (the full system capacity).

A full simulation run (i.e. a simulation that results in the steady-state mean values of TMAX, TAVG and TSTD) is executed under the the hybrid SJF/FIFO policy. For each one of all of the thresholds, i.e. each simulation running, the system has a respective TMAX value in statistical equilibrium. It forms a set of TMAX values that corresponds to the grid from zero to the total number of entities present in a system with the full capacity of 145 entities. It is important to note that, because of the stochastic characteristic of this kind of variables, not all the busy period cycles (a cycle that starts from zero entities in system, passes through

a maximum number of entities and it is concluded by being back to zero entities) of one full simulation run-time reaches the full system capacity. Considering the system that starts out its operation with the SJF scheduling, the goal is to toggle the policies every time the total number of entities equals the threshold value and, at the end of the simulation run, to record the TMAX, TAVG and TSTD mean results. To illustrate it, consider Fig. 40 with only one threshold value set up to 60 entities. This means that the system keeps no more than 60 entities during the SJF period of the hybrid policy. The complimentary number of entities to finalize the running cycle (that can reach other 85 entities until a possible full system capacity) enters the servers with FIFO policy. The correspondent mean variables' values for the whole simulation in hybrid scheduling are the values to be recorded. It is expected that the results of the simulation through the full grid of the threshold values with 2.5000 E of inbound traffic may show 1) the behavior of each variable subjected to the SJF's fluctuation, 2) the mutual compromise between these variables.

Case 2: *Analysis of TMAX, TAVG and TSTD as a function of threshold values and overloading.*

The goal of this case is to evaluate the model's performance subjected to instabilities (traffic overloading) that makes the number of entities surpasses the system capacity. The results are shown and discussed in Section 5.6 (Figs.B and 50). The comparison with the non-overloaded system is also one key motivation. The system is overloaded when the input traffic is excessive to the maximum designed service capacity, a condition that may be temporary, intermittent, or permanent (Fig. 47). The number of entities monotonically increases until the system reaches a sufficiently large number of entities. In any of the cases, the hybrid model performs a policy change, starts the inflow period and it is not able to leave the inflow period to enter the outflow. The best case of a system under overloading is when it naturally rapidly moves to the outflow. During overloading, the DES implementation of the proposed model performs the switching of policy in a cycle of one threshold period. This routine runs for each threshold period until the end of the run time, or until the system goes naturally to the threshold's value in an outflow period. The model is also able to be adapted to different periods for each policy.

Case 3: *The Conceptual Surface Map.*

This surface map of previously estimated variable's responses (via DES) is designed as a 3D graph of a visual tool to evaluate and manage the performance of systems with partially (or totally) unknown results. Each point in the surface map is a steady-state value of the observable variable got through a full simulation run. The observable variable's estimated-values vary either through the number of entities in system, or through the traffic intensity,

or through both (Section 5.6.2.4).

In a system which the controlled variables are online followed, as for example, through the identification of the total number of entities via RFID, the map enables an analyst/manager graphically and previously estimate (via DES) bottlenecks in a future perspective. This applicability allows the professional to change an ongoing behavior with a management action to be made in the neighborhood of a present value of the controlled variable. This visual characteristic can anticipate actions that may be taken to change course (trajectories) of an ongoing operation. This mapping applicability goes beyond long-term planning to the management supported by a performance map of a variable.

5.6 Results of the Simulation Model

This section shows two types of results: 1) the verification's results presents data, figures and analyses of the model engine, in terms of how the model works within the system's requirements, i.e. how it change policies during floating traffic; 2) and the application results for the described cases, e.g., mean-times results and responses to changing parameters.

5.6.1 Results of the Verification Process

This section shows results to the model of three parallel servers described in Section 5.3.2. The toggling process (Fig. 47) was set with one unique threshold value, the in_THRESHOLD and the out_THRESHOLD values are the same, in a simpler implementation compared to Fig. 40 with the sense of the *total_THRESHOLD* in Fig.41. None of the tests presented typical problems of e.g., run time or modeling.

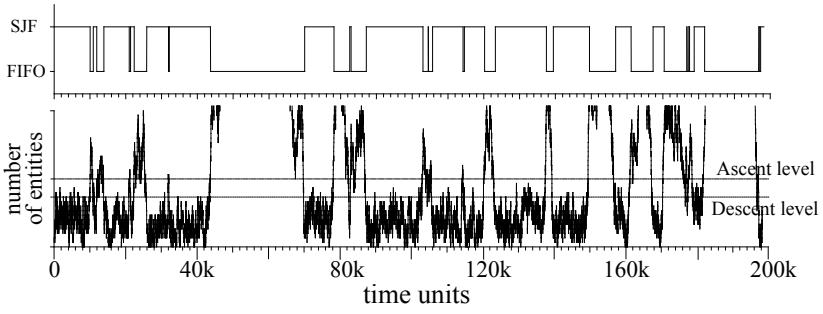


Figura 47 – Case 1.1: Signal Response of the Changing Policy Engine. Lower graphic shows the 'Upper' and 'Lower' levels of the total number of entities in system. These levels trigger the scheduling policies as shown in the upper graphic.

Differently from the parameters used in the case studies, Fig.47 shows one example of the different verification's experiments (the longer one ran 1,000,000 ut). They took place to

show the operational results applied to a longer run time compared to the cases' run time. The reason of the longer run time is to evaluate possible crashes of the model. The discontinuity present in the *time units* axes (from 40k to ≈ 70k) of the Fig. 47 has two interpretations. The first is that the used scale of the vertical axes was not enough to represent the total variation of entities. The second analysis see the gap as the result of an overflow threshold for the set of system's parameters in use. Considering this view, beyond this threshold the system may crash, or it may be out of a typical requirement, e.g. TMAX. The inflow threshold value is identified in the Fig. 47 with the label *Ascent level* and the outflow threshold, with the label *Descent level*. The upside curve of Fig. 47 shows the policy toggling. In the lower graphic it is possible to see the number of entities in the execution-time line. However, as later discussed, some times the traffic is so intense that the outflow takes a long time to happens, or it does not exists.

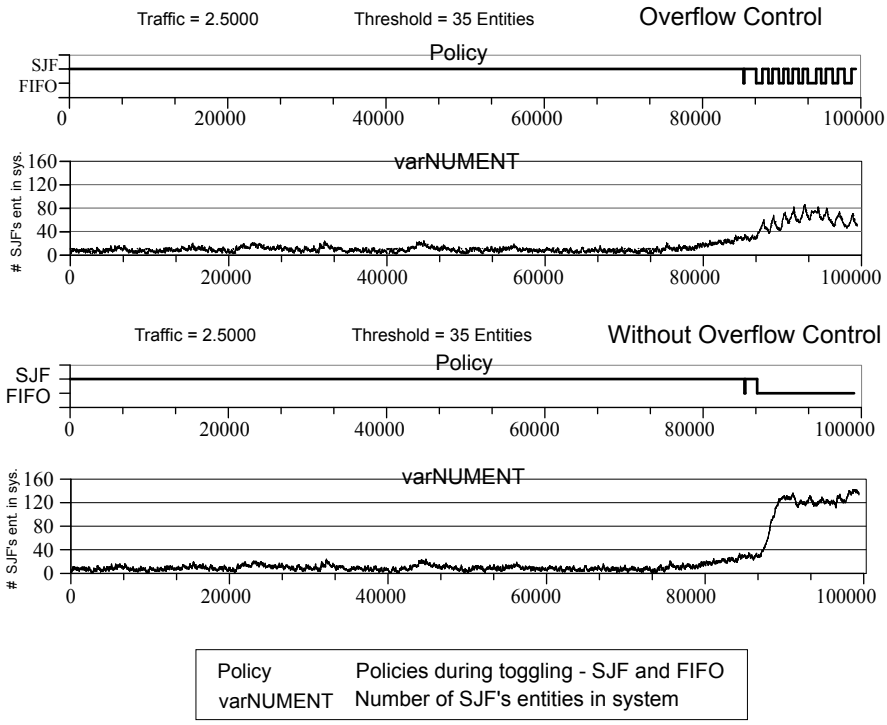


Figura 48 – Case 2.3: Toggling policies during Overflow period.

Fig. 48 shows the simulation model in two operational conditions: to toggle during the overflow (the two upper graphs labled *Overflow Control*) and, do not toggle during the overflow (the two lower graphs labled *Without Overflow Control*). The model performs as described in Section 5.3.2 in the presence of high traffic ($\rho \geq 1$). The first part of both overflow conditions, represented by the timeline from 0 to 83.000 ut, shows the same performance. From this timeline point the *Without Overflow Control* graphs show *varNUMENT* values greater than 100 entities and the *Policy* graph stays is FIFO policy until the end of the

simulation. It means the system reaches its maximum service capacity and could not leave it. Differently, the Policy - Overflow Control graphs show the toggling police (upper *Policy* graph) from 83.000 to 100.000 ut. The variable varNUMENT responses identify a system not in overflow with its peak value around 80 entities, far from the system limit of 145 entities.

Sensitive analyses are used to observe how the model responds to the planning conditions, considering the variables TAVG, TMAX and TSTD subjected to the complexity of the hybrid policy when subjected to instabilities. This is the case, e.g. of the traffic's overflow shown in Fig.48.

5.6.2 Sensitive Analyses for the Hybrid SJF/FIFO Policy

These case studies evaluate the TAVG, TMAX and TSTD variables through the variation of the total number of entities in the steady-state system. Each point of the next graphs corresponds to one complete simulation run. Recall that TAVG, TMAX and TSTD values are obtained through the simulation model. These variables are totally application-dependent, especially the TMAX value that is obtained with each new replication. For example, a system that operates uninterruptedly for 24 hours will have a certain value of TMAX. A system that operates uninterruptedly for five days will have another TMAX value. If the largest job arrives at the system during the first minutes, considering the first case it should leave it 24 hours later, and in the second case, it would take 5 days for leaving it. The simulations depend on long run-times and large number of replications to give answers with accuracy. One of the simulation's objectives is to identify the system's worst-cases. This is the case of TAVG and other performance indicators. The modeling parameters are central to the work's success.

As earlier mentioned, we know that the SJF discipline is the one that results the smallest *TAVG* among non-preemptive scheduling policies even in the M/G/1 queues. Therefore, it is also the one that allows the greatest TMAX. In turn, the FIFO discipline that makes it possible to the lowest *TMAX* among non-preemptive disciplines as seen later.

5.6.2.1 Results for TAVG and TMAX Without the Overflow Toggler (Cases 2.1 and 2.2)

The results of TAVG and TMAX from case study 1 are shown in Fig. B A and B, where A is run for a traffic of 2.5000 E and B is 2.3684 E, respectively.

In Fig. Ba, each x-axes' coordinate represents a single threshold (i.e. in_threshold = out_threshold) of the hybrid scheduling algorithm. The algorithms starts out as initial condition under the SJF policy. For example, if the threshold is 60 (i.e. x = 60), TAVG and

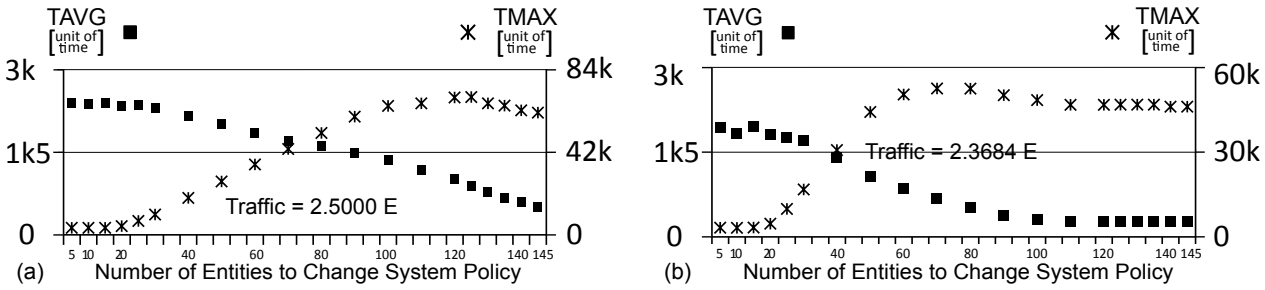


Figura 49 – Case 1.1: Results for *TAVG* and *TMAX*. Sync variables' behavior through the number of entities subjected to SJF: a) Traffic = 2.5000 E, b) Traffic = 2.3684 E

TMAX are respectively the steady-state performance values obtained for the simulation run when the system executes and change policies under this threshold value.

Figure B shows a first range (which changes with traffic) in which the variables are not mutually compromised, or the variables are uncorrelated. This range represents predominance of the FIFO policy in the system (e.g., with 2.5000 E, are approximately 20 entities in SJF for 125 FIFO, or a close ratio of 85 % FIFO). Then, after point 20 during the progression of Threshold values there is another interval in which TAVG and TMAX exhibit trade-off, or negative compromise, to the maximum point of the TMAX values. In other words, while TAVG is reduced, the TMAX values increase. This relationship changes from the maximum point of TMAX, where the values of both TAVG and TMAX reduce. This interval that represents a positive commitment between the variables follows until the system is saturated, in this case study, with predominance for the SJF policy. Note that the increase in traffic intensity lengthens the trade-off state in relation to the threshold increase.

As the number of entities subject to interference increased (for various traffic), Fig. 50 showed that $TMAX_{SJF}$ can reach, e.g in high traffic (2.5000 E) $\sim$ 70k ut compared to $TMAX_{FIFO}$ ($\sim$ 500 ut). In this way, if there is no restriction regarding the mean maximum residence time, the service discipline should always be SJF. This means that if the TMAX values still within the stated requirements there is no need for changes. Having considered that, justified our adoption of a hybrid policy that started the system by searching for the lowest TAVG when in SJF and changed to $FIFO$ aiming to limit TMAX.

5.6.2.2 Results for TSTD and TMAX Without the Overflow Toggler

The model also shows the mean results for TSTD. It is relevant to observe these values together with TMAX' values because of their similar behavior through the number of entities subjected to SJF as Fig. 50 shows.

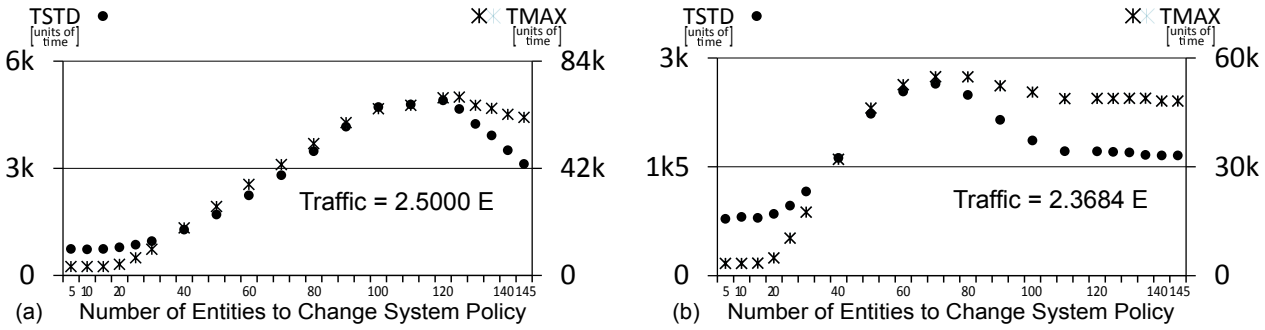


Figura 50 – Case 1.2: Results for $TSTD$ and $TMAX$. Sync variables' behavior through the number of entities subject to SJF.

It is easy to see the positive compromise between both TMAX and TSTD. The same behavior appears in Figs. A and B. There are three remarkable regions: the first one at the left and at the right sides of the graphs - these regions are remarkably shaped as horizontal lines, that represents the variables are not compromised to each other; the second region has aligned points according to a Sigmoidal shape (an ascent monotonic shape). These monotonic behaviors is sync and pass each one through a maximum point to begin a descent behavior. The third region is this descent alignment points as an S-curve shape that becomes stabilized (the first region at right side). The regions Sigmoidal and S-curve represent the positive-compromising variables. They are significantly positive to be explored as mutual performance-indicators.

TMAX and TSTD behavior has those intringuing hump or crest followed by a stabilized horizontal line. Recall that a permanent overloading is similar to this sigmoidal curve. This similarity is the motivation to the next approach for studying a proposed model that considers overloading.

5.6.2.3 Results for TAVG and TMAX With the Overflow Toggler

The model approach for overloading is later described in Section 5.5. Here are presented the results of the simulations applied to the hybrid policy and subjected to increasing traffic as shown in Figs. 51 and 52. To recall the policy toggler, please see Section 5.4.1.

The performance's differences from toggling and do not toggling during the overload period subjected to the same traffic and the same parameters are huge as seen in Fig. 51. The TAVG falls from about 2600ut to around 700ut in an operation mainly with FIFO (a reduction around 70%), and in the opposite side of the curve with TAVG mainly in SJF the performance was improved from about 570ut to around 260ut (an improvement of about 50%). The worst case of TAVG using the overload toggler, i.e. during 100% FIFO policy (=

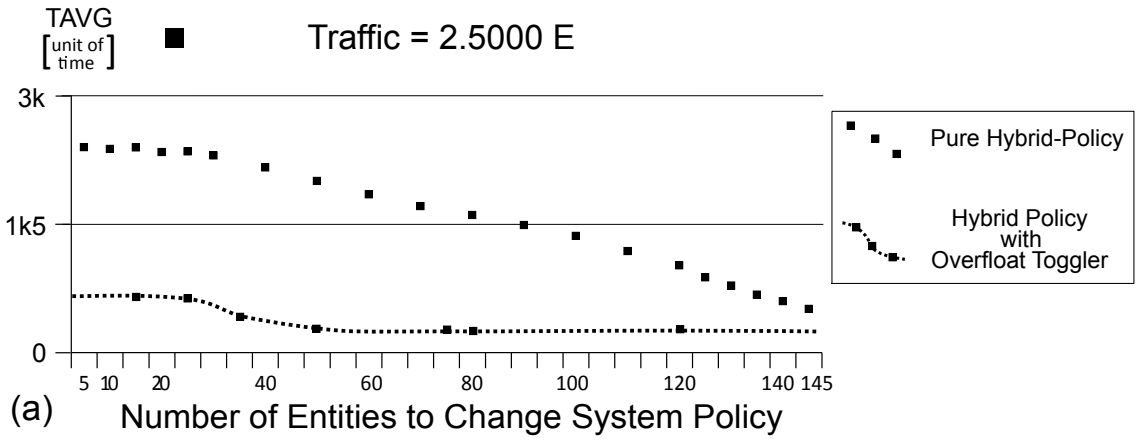


Figura 51 – Case 2.1: Results for *TAVG* with Overload Toggler. Comparison analysis of the TAVG between results with and without the Overflow Toggler.

700ut) is less than 20% higher than the best TAVG performance in SJF without the overload toggler.

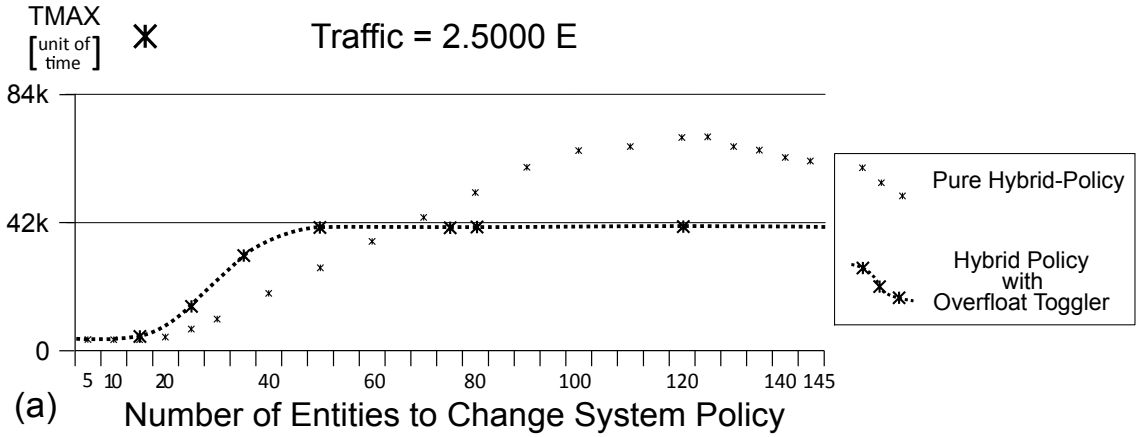


Figura 52 – Case 2.2: Results for *TAVG* with Overload Toggler. Comparison analysis of the TAVG between results with and without the Overflow Toggler.

The performance's differences from toggling and do not toggling during the overload period subjected to the same traffic and the same parameters are relatively important as seen in Fig. 52. The TMAX falls from a peak value of around 70000ut in an operation mainly with SJF to about 40000ut (a reduction greater than 40%) with the overload toggler. This highest value of 40000ut stabilizes with around 50 entities submitted to SJF against an operation fully with no stabilization without the overload toggler. It is also important to note that thee is no crest in the response's curve during the use of the model with the policies' toggler applied to the overload period.

These results may be used to support a planning strategy based in a performance

mapping detailed in the next sections.

5.6.2.4 Maps of Performance Analyses

The mapping method is expected to evaluate and manage the performance of systems characterized by results that are partially (or totally) unknown. The map enables an analyst/manager to also study the behavior of a given variable over time as a future perspective. As a variable's behavior we understand, e.g. the variable's values that either vary through the number of entities in system, or through the traffic intensity, or through both (a 3D graph). The mapping extends the applicability of the discrete simulation. When variables' values are graphically describing one system's behavior (Fig. 53), it is visually easy to predict results that were previously estimated (via DES) in the neighborhood of a present value. This visual characteristic can anticipate actions that may be taken to change course (trajectories) of an ongoing operation. This mapping applicability goes beyond long-term planning. Figure 53 is a hypothetical example of a 3D graph to illustrate a performance map of a variable.

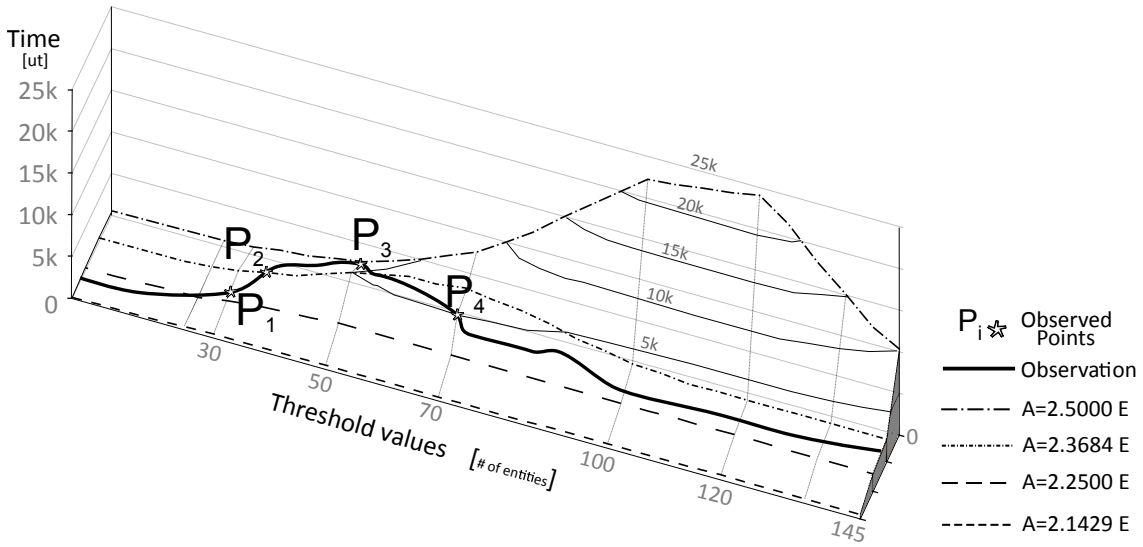


Figura 53 – Case 3: Hypothetical 3D-map of a system variable. The time-based variable's values varies in the y-exes through Traffic values in the z-axes and through Threshold values in the x-axes.

This hypothetical 3D map is a kind of geodesic map of graded *surface-lines*. The values of the grade are the *Time-line* values. One example is the surface line labeled with the level *5k ut* where all the points represent the same value *5k ut* of the Variable. This line has two flagged points P_3 and P_4 in different positions. P_3 is lined up with the value *50 entities* of the *Threshold-axes*, and differently, P_4 is lined up with the value *70 entities* of the same *Threshold-axes*. The points P_3 and P_4 are also aligned with values of the *Traffic-axes*: P_3 is near the 2.5000 E value, and P_4 is in between the values 2.3684 E and 2.2500 E. Consequently,

each point has three coordinates (Threshold, Variable, Traffic) that completely identifies it in the map.

There are four dashed lines on the map, each representing a given amount of traffic. A line is constructed by defining the corresponding traffic value (in the DES model) and executing the simulation routine for all x-axis 'Threshold' values. The model performs a complete simulation at each point and stores the corresponding value of the variable. The complete sequence of all points is represented by the corresponding dashed line. The synchronization of the four rows by the corresponding Threshold values of the x-axis results in the response surface of a Performance Map. The process can be repeated for as many variables as are of interest in the evaluation and/or management of the system.

The highlighted line, which is marked with points P_1, P_2, P_3, and P_4 represents a hypothetical performance monitoring during a given system management period. This hypothetical set of successive values, which are measured for the Variable, forms this line denoted *Path* of the Variable. The trajectory extends to the P_1 point where the system reaches the traffic value $A = 2.2500E$ (large-dashed line) after the increase of 30 entities in SJF (Threshold axis). From P_1 to P_2 there is an increase of less than 5 entities on the x-axis, so that the system changes to the $A = 2.3684\ E$ level. This rapid increase modifies because the observed result changes the system to the last level (tracer-dotted line A = 2.5000 E) at the point P_3, but more slowly with the occurrence of another 15 entities in the mark of 50 entities of the Threshold axis. The last point highlighted is P_4, where the system loads with 20 more entities (mark 70 of the Threshold axis), however with reduction of traffic to a value close to $2.2500\ E$ (wide dash line). It may be assumed in this hypothetical example that there may have been managerial action for the system not to enter the region of extreme values (mountain peak form on the map) of the Measured Variable.

This comprehensiveness of the mapping method responds with a new perspective of solution to problems with unpredictability and complexity that, when approached by other methods, receive unsatisfactory answers. This, for example, is the case of hyper-heuristics, whose boundaries were highlighted in (BRANKE *et al.*, 2016). Another example was the difficulty in evaluating the SJF variance by comparing the formal model proposed by (SANDMANN, 2013) in high traffic conditions ($\rho > 0.8$), where its calculated values diverge of the Monte Carlo method.

5.7 Discussion

The proposed DES model is built with commercial software, which is the case of *Arena* ®, as it seeks to have general applicability regardless of the application area of the

case studies. In this context, the alternation between FIFO and SJF policies, which we call the *Hybrid Policy*, first of all, responds to general application, since it is the most intuitive and well-known service policies. Within the conceptual context, these policies allow competition (trade-off) between its main characteristics: the best average residence time (TAVG) in favor of SJF and the best average maximum time (TMAX) in favor of FIFO. Because the competition is possible only if the jobs are correctly addressed to the servers, all the jobs are addressed by demand in a dynamic queue. This is a pull model driven by a kanban-type logic. Thus, the model performs the dynamic management of the alternation between the policies according to the number of jobs in the system. In this way, e.g., the model allows the reduction of the TAVG by the inherent characteristic of SJF, with the FIFO policy adding significant control over TMAX. The control of TMAX implies a reduction of the mean variance, which, in turn, also contributes to the reduction of TAVG.

The implementation of the DES model with hybrid SJF/FIFO policy has not yet been described as far as our research can achieve. The approach of this model to external traffics ($\rho = \lambda/\mu \; and 1$) is also not explored. The use of a simulation model for operations management is also an unexplored issue. This set of characteristics allows us to affirm that the proposed model is a novelty. With the results achieved, the model allows performance targets that are relatively difficult to achieve, thus increasing its applicability. The approach described for the model allows the planning of systems with complexity similar to the one we describe in this work. It also allows the mapping and management in spite of mathematical formalism. The final model depends only on a basic, validated initial model.

Concerning to the system performance, Sandmann (2006) show that, for single server systems with hybrid policy at a 50 % to SJF / FIFO toggle rate, loss for $\rho = 0.9$ occurs. Although it is significant, this loss can be disregarded, taking into account the context of benefits. In the dynamic hybrid policy model presented, Figs. 51 and 52 show a significant performance of both TMAX and TAVG. Additionally Fig. 54 shows the delay of the losses when the toggler is used during overflow. For example, if in a regular operation the delay begins after 10 hours with a certain traffic, with the same traffic and the use of the toggler during overflow, the delays will begin after 20 hours. The system gains time to operate without overflow, i.e. it gains extra temporary flow.

The delay does not guarantee reduction of losses. However, shorter operations tend to benefit from this feature. For a 100000 ut runtime, the average time analyzed by the graph *Toggler with Overflow* of the figure will be lower when compared to the graph above it. This feature enhances the operations with shorter times.

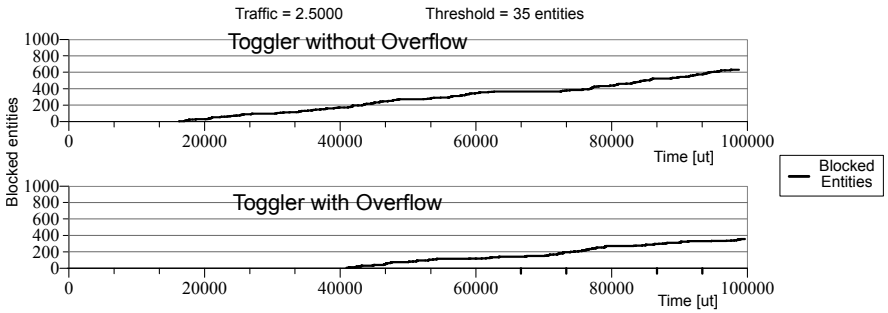


Figura 54 – Case 3: Comparison between the number of blocks with and without the Toggler_overflow.

5.8 Conclusion

In this work, we offered an approach to hybrid scheduling of jobs that aims to maintain the system performance at pre-defined levels. Two performance variables and three case studies were chosen to illustrate the approach (i.e. mean residence time and mean maximum resident time). The model changes policies according to thresholds and the number of entities in the system in an attempt to conform to the requirements.

In the case of a simple baseline model that evolves by incremental validation, the mapping method allows the description of responses, e.g, in the form of traditional graphs and surfaces that map the behavior of system variables as a function of one (or more than one) given control variable. The choice of this variable was due to its ease of observation and applicability to the underlined problem. This way of including complexities is what allows the mapping method not to be limited to algebraic, a-priory known models. The mapping method is a planning tool that allows analysts and managers to identify a set of the system's characteristics.

We illustrate the mapping method showing how to build it, an also the way to use in an actual application. A real-world application on following a process or a daily operation is a management application. Sensors and interfaces can collect data to feed information to a platform where a management team can make decisions and take action based on the set of estimated information in the responses map this model generates.

2) Can the model be used as a management support tool? 3) Does the model ensure the maintenance of a certain established QoS requirement? For example, consider an application that needs to keep a given TAVG limited by a maximum value to satisfy the system QoS. 4) Can the model estimates a change in the upward trajectory of the TMAX to reduce it to a level below a certain value (i.e. a set point defined by the application)? Can the model keep the system QoS within the established requirements?

The work on the the hybrid online scheduling approach opens up several lines of work for future research. For example, one possibility would be the selective control of entities, since it allows the treatment of entities in an individualized way (in terms of characteristics or attribute values) by extending the approach to include the mean time. A practical example is the need for control over "rats and elephant´´ (relatively very small and / or very large entities) in associated queue problems with the technologies applied to WEB IoT and Cloud. In this way, the proposed method allows, in a simple way, the user's action, for example, to switch the discipline (SJF / FIFO) according to the number of entities present in the system, in order to obtain a trade-off according to the importance (the weight) given to your requirements of interest.

The approach could be extended to include a predictive model of trajectory change, and for the time being it is limited to the use of a known analytic model, or a real data set large enough to effect of the validation of the simulation model. Also, the increment of an automated plot of the results of observable variables. This model could be tested for different types of probability distributions. Another future approach would be to make the model adaptive, so that it can, automatically, change its mode of operation. Predictors were used to evaluate the number of units in the system like Kalman filter and the rate of arrival of new requests. Through Little's theorem, for example, it would be possible to foresee the value of the delay to which users would be subjected and to act in advance on the discipline change (SJF / FIFO). In addition, to make the system even more adaptive, it is possible to incorporate a procedure using Fuzzy logic for decision making.

Another interesting possible future work would be to study how the model could be adapted and configured for a real-world waiting scenario, e.g. banking waiting area. In such modern systems, the customer, upon arrival in the system, interacts with an online terminal to define the type of customer and service, which may be approximated to the size of job. As mentioned earlier, current scheduling is mostly manual and it neither efficiently estimates and accounts for overall system efficiency nor it meets fairness requirements, which are features that may be processed through the proposed model.

The model introduced is an important step towards improving overall system performance. Two essential variables (TMAX and TAVG) are affected by the dual-scheduling approach. However, real world system may also exhibit jobs that have a deadline, which may be highly critical or not. The scheduling approaches that cater for deadlines are the ones commonly found in the domain of real-time systems. One of the scheduling approaches that stand out in this field is the so-called Earliest-Deadline-First (EDF). While EDF factors the deadline of a job but does not consider throughput and average behavior, FIFO and SJF do so by nevertheless ignore the deadline of a job. Therefore, an approach that combines and

toggles among FIFO, SJF and EDF, in an attempt to conciliate TMAX, TAVG and deadline (and keep their values bounded), may be a strong candidate for future research work.

Appendix - Guidelines

To build the simulation model, it is necessary to consider the following guidelines:

1. Begin with a simple and validated model (e.g. the JQN in FIFO scheduling);
2. Explore a given interference variable (e.g. the SJF policy) introduced by incremental module(s);
3. Set the capacity limit of the new validated system (e.g., total capacity of 100 entities in the system);
4. Allow different traffic intensities;
5. Increase the interference (i.e. the number of jobs admitted during the SJF policy), in fixed steps (e.g. each 5 entities in the system) in a complementary way to the system's capacity (capacity = 100, SJF = 20, FIFO = 80);
6. Ensure flexibility for the interference period, i.e the interference period can start at any point in the system's inflow/outflow cycle;
7. Ensuring elasticity for the interference period. Usually the system starts at a point in the inflow period and ends at the same point during the outflow period. Differently, to ensure a model interference's elasticity means that independently of the cycle, the model needs to ensure flexible ranges of cutoff. In other words, the change of policy starts always at a given point of the inflow and finalizes at a different point during the outflow. For example, the SJF policy begins with 20 entities in the system during the inflow period (while the number of entities rises up before reaching a local maximum, e.g. 26 entities) and returns to FIFO by reaching 16 entities during the outflow period (after passing through the local maximum);
8. Ensure the integrity of the interference. The cutoff values are always the same in different inflow/outflow cycles (unless there is a change of parameters);
9. Ensure that all servers respond in sync to the current policy;
10. Ensure that all jobs in a queue are subject to the same valid policy immediately after a change;

11. Classify observable system variables into different classes by job sizes;

12. Map the changing of system variables.

Appendix - Activity Model

The proposed model may be represented through an activity diagram, as shown in Fig. 55. It may be structured around *Product Processes, Queuing Processes* and *Information Processes*. The physical stream (represented by solid lines) distinguishes and route the activities of wait-for-attendance (*Queuing Process*) from the activities of service (*Products Process*). It is important to note that an entity has the same chance of being served by any server. In other words, the first server to serve a job of a given entity can be *Server 1*, or *Server 2*, or *Server 3*. This perspective is presented in the *Product Processes* column. Dashed lines distinguish and route the flow of data in the system (named *Information Process*) passing through the model.

The Information Processes column starts with a connection of the upper horizontal bar to Servers identified by 1, 2 and 3. At the end of information flow (the lower horizontal bars), the information processes divert to the Servers in order of the entity's attendance (FIRST, SECOND or THIRD), not by the server identification (1, 2 or 3). The *FIRST Server* label means the first of the three entities' products to be loaded. This label is unique to those entities that came from outside the system and passed through none of the servers, in the sense of exogenous requests. The reetrant entities are identified by the *SECOND Server and THIRD Server* labels.

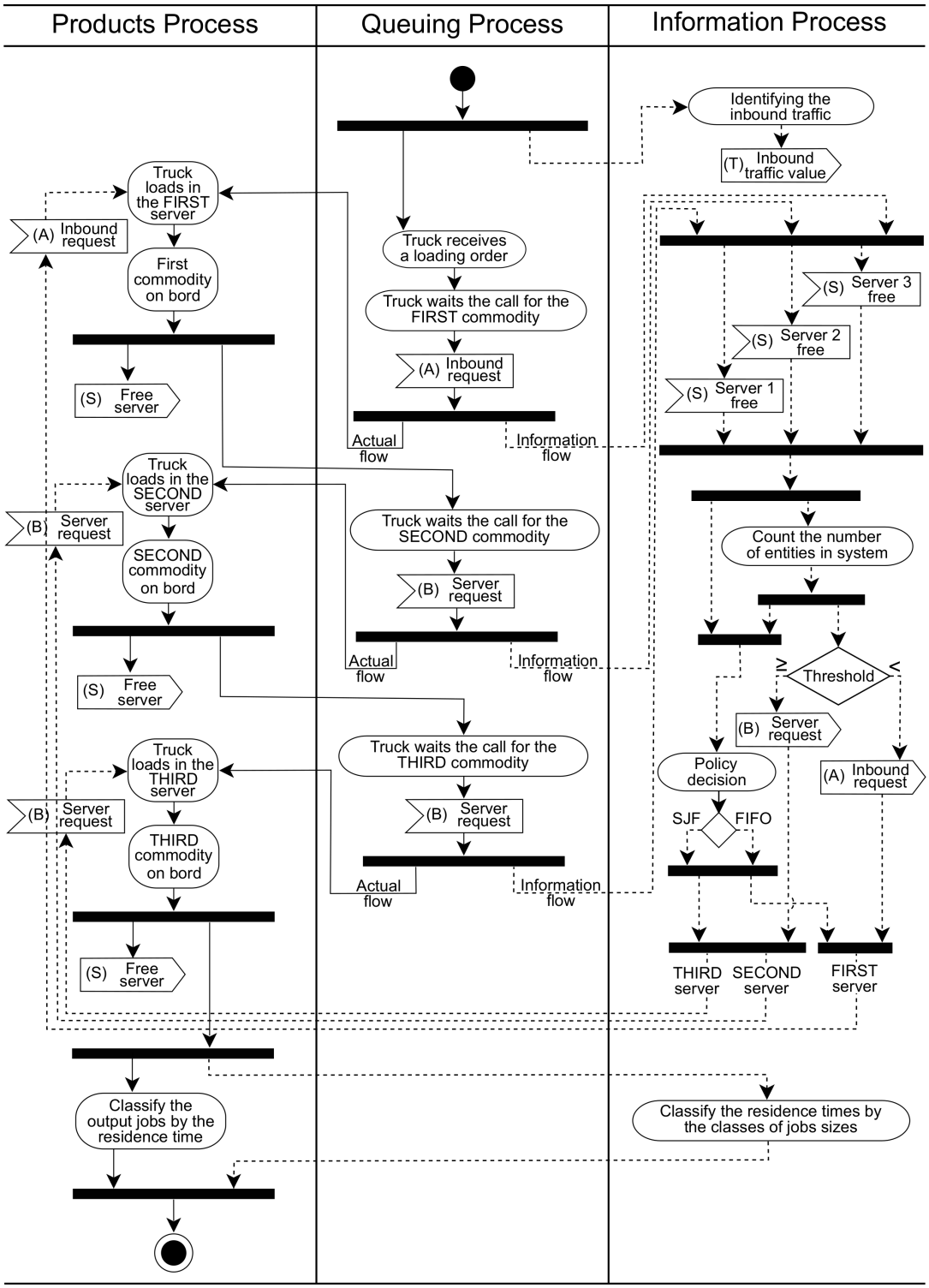


Figura 55 – The activities and relations between the physical and informational processes of the simulation model.

6 Identifying Jobs' Sizes Performance in the Hybrid SJF/FIFO System

We know that the size of jobs can negatively impact the performance of queuing systems. This can happen both due to the number of relatively small jobs causing high latency, as well as for jobs with relatively high sizes. These negative impact are responsible for performance delays and reductions in system flow. These kind of loss of performance, also present in systems of greater complexity, are usually associated to the largest jobs' sizes present in the tail of the service times distribution. We already know that those losses have important association to jobs' sizes around the statistical mode. In systems under more complex scheduling policies, and the *Selective Hybrid Policy* belongs to them, we know that intermediate-size jobs can be identified using *Continuous/Discrete Hybrid Simulation* (Chapter 4). Differently, this chapter shows another method based on the classification of jobs in two levels to be performed with DES. It identifies the job size that reduces the system performance.

6.1 Introduction

Knowing the tasks' sizes in advance of processing is an advantage for building strategies and performance improvements in queue systems (SANTOS *et al.*, 2018). It allows, e.g. the use of the shortest-job-first (SJF) scheduling. However, it is still necessary to manage the size of jobs, e.g. when there are customers in the supermarket cashier's queue with their small grocery shopping, but behind a customer with four crowded shopping carts. Differently, Sun *et al.* (2016) give an example of performance issues about relatively small jobs. They show the influence of small tasks on real-time processes that require ultra-low latency. Another case is cited by Serwadda e Phoha (2015) in a work on information security during the issues of long-tail distributions. Jobs with relatively high sizes are presented by Mor (2010) for causing delays and reducing the flow of queue systems. The same author shows that intermediate-size jobs can lead to the performance reduction of single-server systems submitted to SJF policy. Santos *et al.* (2018) show that parallel server systems with SJF/FIFO hybrid policy can also loose performance because of intermediate size jobs. A relevant issue is the identification of these jobs. Chapter 4 of this thesis showed that intermediate-size jobs, subjected to systems with a SJF/FIFO hybrid policy and traffic fluctuations, can be identified using discrete/continuous hybrid simulation through the concept of job's predictability.

The strategy we use in this work is to identify the classes of jobs that most delay the

system ($\%B_{class}$). It is based on the relation of the sum of service times of the K jobs of that class ($\sum_{j=1}^{K} B_{j\_class}$) to the sum of service times of the jobs of all classes (B_{total}).

$$\%B_{class} = \frac{\sum_{j=1}^{K} B_{j\_class}}{B_{total}}, \qquad j = 1, 2, 3, ..., K \tag{6.1}$$

We calculate B_{j_class} as the product of the K number of jobs in the class and the T_{class} value of the service time that identifies the class, e.g. a class of service times from 10 ut to 15 ut is identified by its upper value, i.e, it is the 15 ut class, and Eq.6.1 changes to:

$$\%B_{class} = \frac{\sum_{j=1}^{K} (K * T_{class})}{B_{total}}, \qquad j = 1, 2, 3, ..., K \tag{6.2}$$

The $\%B_{class}$ values are shown in Section 6.6.

6.2 Related Work

Beemsterboer *et al.* (2017) apply a hybrid SJF/FIFO model in a make-to-stock (MTS)/(MTO) make-to-order method for stock-control of an enterprise. The MTS usually attends orders with FIFO scheduling, and the MTO with the SJF policy. Results show the better performance for hybrid scheduling compared to the priority scheduling, but reported issues in relation to the implementation and applicability costs. The authors did not use jobs' size control in the sense of individual/class of sizes. Our proposed method use the job's identification to manage the system through a path of estimated responses (built a-priori) in order to deflect this system from an estimated bottleneck.

Elmougy *et al.* (2017) study system design issues applied to cloud computing infrastructures and application services. They designed a hybrid algorithm named SJF-RR-Dynamic-Quantum, SRDQ to minimize the starvation dilemma in complex computing and big-scale. Differently focus on a simulation solution related to a general problem, for example, as in services, manufacturing, production or transportation processes. Our work proposes continuous simulation associated to discrete-event simulation to take advantage of the fairness metric. It is proposed to understand the unknown behavior of the system submitted to traffic instabilities during the SJF/FIFO policy. Our work looks for improvements of the expected results of regular delay metrics.

Wierman *et al.* (2007) analyze the first and second moments of the SJF's residence time according to the expected maximum duration parameter depending on the service length. We associate this approach to the collective approach of discrete simulation results, in terms of system mean values applied to actual situations. It estimates the job's sizes that

can cause the largest delay in a system with the complexity of a hybrid SJF/FIFO policy submitted to varying traffic and overloading.

Sandmann (2013) addresses the fairness issue regarding service disciplines. Our work is also related to this issue. Differently, we analyze its limits depending on different offered traffics to extend them into a map of estimated responses which guide an analyst to manage an online operation.

An airport's application is implemented by Wu e Xie (2017) to the load balancing policies for an airport baggage handling system. The authors considered the difficulty and accuracy of mathematical formations to model a discrete-event simulation solution. They develop a platform with a proposed hybrid round-robin (RR)/(JSQ) joint-shortest-queuing policy. They show the solution reduces waiting times and increased maximum achievable system throughput. Our method allows the system's planning as a long-term application and a system's managing that offers the possibility of a short-term application.

Looking for better performance on WEB or online applications, Büke e Chen (2015) tested the number of times a service request of one set of a parallel servers matches the availability time of a server in a set of parallel servers. The authors identified that, if no control mechanism is employed to provide better performance, these systems are unstable for any set of parameters. They prove that the rejection rate may be an increasing function of the matching probability, and also prove insensitive to results related to the average queue lengths and waiting times. We do suggest our model as an IoT/WEB application and, differently from the Büke e Chen (2015)'s objective, our method does not intend to match job's size to server's availability, but identify the job's size that effectively drive the system for the largest delay on a better fairness solution.

6.3 The Proposed Model for the Job Identification

This model describes the *Classifying* module that ends the simulation model of Chapter 5. The classifying function identifies the jobs' size that negatively impact the system performance, which means either residence times greater than those performed on the FIFO model (a longer tail), or the same classes but more crowded than the FIFO's tail. Both longer tail or more crowded tail mean the existence of entities delayed when compared to the equivalent performance on FIFO's model. *Classifying* provides the jobs a two level ranking (Fig. 56). The first level groups the outbound jobs of the simulation model into different classes of mean residence time (TAVG's classes) in a first ranking level of Groups (e.g., Group 1, Group 2 and Group 3). The next level selects one group to evaluate which are its job sizes (service times). This information enables the analyst to manage the system's responses on a

job-size basis that do not depends on the server (A, B or C) it was performed. In this way, the manager is able to add control to the entities whose job sizes have a negative impact on the required results. There are two important processes from the *Classifying* module, pedagogically separated in 8 activities, each one understood as an action with beginning and end (Fig. 56).

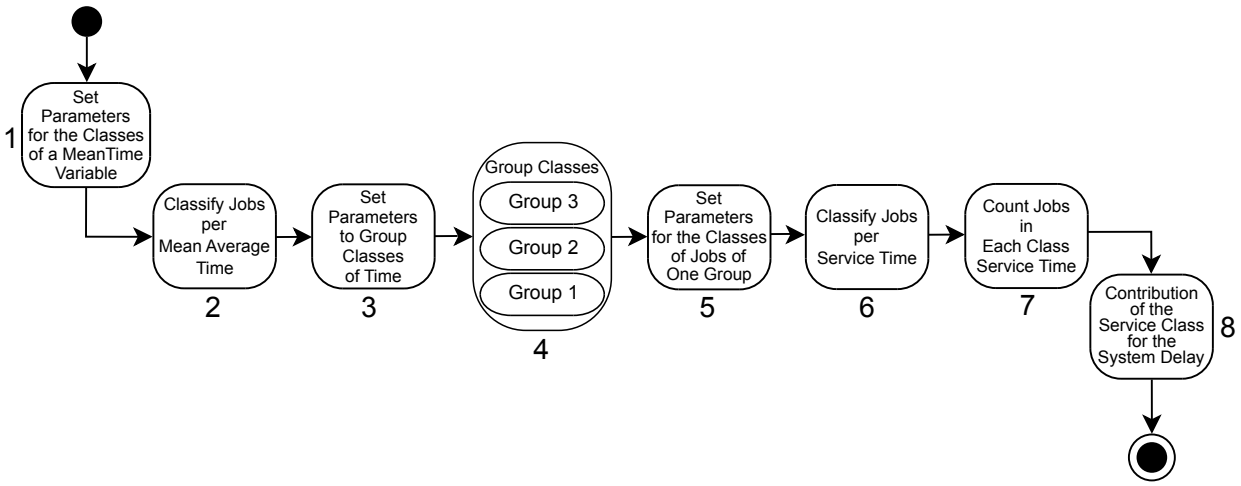


Figura 56 – Activity diagram for the Classifying module of the Hybrid Model

Initially the first process sets the ranking parameters (activity 1) that define, e.g., one class each five *ut*, to be set for one chosen variable (activity 2), for example, a set of classes to TAVG (mean residence time). The activities 3 and 4 set the group parameters for ranking the TAVG's classes of values into groups. As an example, the residence time of Group 1 are set with values smaller or equal to the mean value ($TAVG_{Group1} \leq TAVG$), the residence time values of Group 2 are defined as $TAVG_{Group1} < TAVG_{Group2} \leq 2 * TAVG$, and the Group 3 is established as $TAVG_{Group3} > TAVG_{Group2}$. *Classify jobs per mean average time* is the core of this first ranking level. This is an activity that build a histogram with values of the mean residence times (TAVG). During planning, the system conditions, e.g., traffic intensity and actual service time are crucial to this classification. The better ranking, the most sensitive evaluation for the TAVG histogram that allows the model to identify the distribution profile for the *Group classes of residence time*.

The second process of the *Classifying* module classify the service times to identify their contribution to the system delay. Activities 5, 6 and 7 rank one selected group of residence times into classes of jobs duration and count the number of jobs in each class of service time. These data allows the *Contribution Class Delay* (activity 8) to identify the classes of service time that most delay the system. The *Classify jobs per service time* (activity 6) identifies the distribution of jobs' classes. The objective of this ranking is to identify the distribution of jobs in one specific group of service times, e.g. the most crowded jobs' classes in the tail of

the distribution of the completed entities that leave the system. The eight activity counts the jobs in this class to use this value in Equation 6.2 that gives the class' contribution (in terms of percentage) to the system delay. It is done by a regular manipulation of the Kleinrock's conservation law of delay (KLEINROCK, 1965) and the Little's law (LITTLE, 1961) for steady-state systems.

$$Delay = N * W \qquad and \qquad TAVG = W + B$$

Where N is the mean number of system outbound entities; W represents the mean waiting time in queue; and B is mean service time in system.

The SJF/FIFO hybrid police is expected to reduce the TAVG value in the presence of the same mean service time B as earlier shown in Chapter 3. This means a shorter W (according to Little (1961)) and supported by Kleinrock (1965), N is expected to increase together with the concentration of jobs around the mode of the residence time distribution. A negative evidence is the presence of a very long tail with a small quantity of entities with huge residence times. This long tail is in the focus of the proposed solution for the job's identification. The implementation's description that follows this section, shows the logical model.

6.4 Implementation

This logical design is built immediately before the jobs leave the system. The simulation model selects the incomplete entities, i.e. those entities that still have a job to be served. The complete entities will be classified per TAVG. This process is the first ranking level. Next, Fig. 57 shows one classifying configuration.

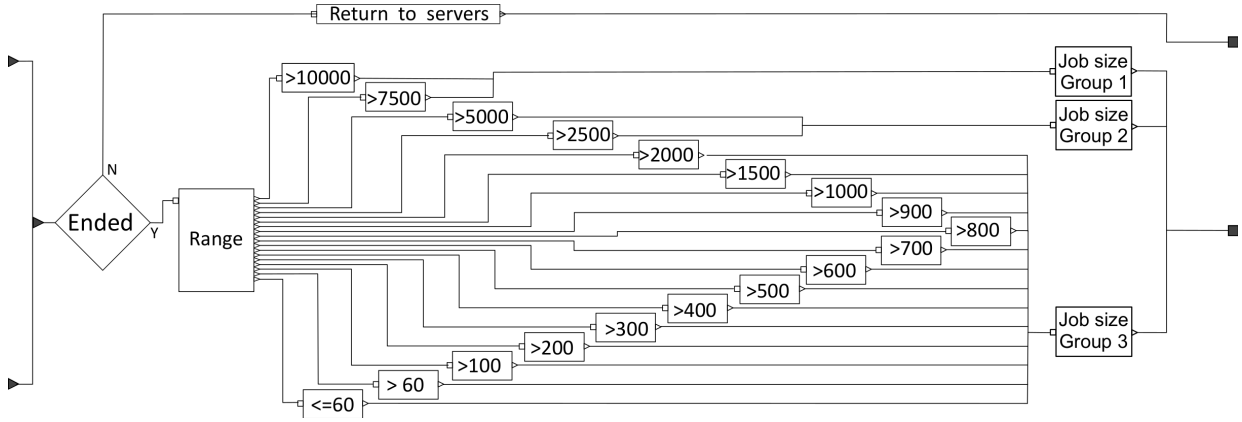


Figura 57 – Classifying/Exit module of the Hybrid Model

Each entity has the *attribute atINTIME* to record the time this entity enters the system. The *Range module* identifies the entity's absolute residence-time, and classify it into the established rank. The processes of the *Job size Group modules*, in this case, use the full classifying data to rank the groups. It means that the model executes a new simulation run to process them, because this proposal does not consider parallel processing yet. Jobs' sizes are recorded in the job's attribute *atSIZE* that is ranked according to the set parameters. Each ranked class is recorded in another attribute (*atCLASIZ*). The values atSIZE and atCLASIZ are used to identify the contribution of the job's classes to the system delay.

6.5 Case Study

The simulation parameters to the case study of the jobs' identification are the same of those used in the simulation model. They are repeated in Table 9 only for the sake of information.

Tabela 9 – Simulation parameters for the job's cases

Replication parameters		Traffic parameters (ut)	
Number of replications	100 units	Inter-arrival time ($I_t = 1/\lambda$)	from 20 to 14
Replication time	100,000 ut	Service time A ($B_A = 1/\mu_A$)	20
Warm-up time	10,000 ut	Service time B ($B_B = 1/\mu_B$)	15
Confidence level	95 %	Service time C ($B_C = 1/\mu_C$)	10
Total system capacity	145 entities		

Table 10 summarizes the case study which goal is to identify the classes of jobs' sizes that cause the largest delays in the tail of the TAVG distribution during system overloading. This means that the goal is to find out the most crowded classes of jobs' sizes in the tail of the TAVG distribution. The system is subjected to the high traffic of 3.2143 E (equivalent to an example of arrival rate $\lambda = 1/14min \sim 4.3$ entities per hour) for the system capacity of 3.0 Erlang in an example of total mean service time of 45 min. This design means a throughput time limited by 15 min ($\frac{45min}{3E}$), i.e. limited to 4.0 entities per hour.

The responses were obtained through a data distribution analyser - the *Output Analyzer* of the *Arena*® simulation package. The *Output Analyzer* ran with the simulation output data for the chosen TAVG group that represents the tail's distribution (Section 6.6).

Tabela 10 – Case Study for Traffic = 3.2143 E ($\lambda = 1/14ut$)

Class	Class Width (Goal)	Classifying condition
Time Average	100 ut 500 ut 2500 ut	TAVG $<$ 1000 ut 1000 ut $\leq$ TAVG $\leq$ 2500 ut TAVG $>$ 2500 ut
Group TAVG	Group 3 Group 2 Group 1	TAVG $<$ 2500 ut 2500 ut $\leq$ TAVG $\leq$ 7500 ut TAVG $>$ 7500 ut
Service Time	Distribution (histog.) Range (histog.) Mean Time (circle)	5 ut (0,40ut), [40,80ut], (80ut,+∞) A=20ut, B=15ut, C=10ut

6.6 Results of the Classifying Process

It is important to recall that the *Overflow Module* of the simulation model is disconnected for the sake of this case study. Furthermore, the largest mean value of a job's size in this work is 20 ut (related to Server A). This system with three servers, maximum capacity of 3.0000 E, and total mean service time of 45 ut (Server A + Server B + Server C) is subjected to high inbound traffic of 3.2143 E ($\lambda = 1/14ut$) and 100% SJF policy. Each one of the Figures 58, 59 and 60 represents a group of averages from the TAVG's distribution (Group 1, Group 2 and Group 3 from Table 10). Each Group shows the distribution of the jobs' contribution to the total delay of the system vs the job's sizes (i.e., through classes of service times). It is important to recall that each server attends one unique type of jobs (Server A to mean service time 20 ut, Server B to 15 ut and Server C to 10 ut). Each figure has three graphs with a histogram at the left side. This histogram represents the general distribution of the jobs' delay contribution to the system's delay. The graph in the middle position is also a histogram that represents three ranks of jobs' sizes: the first one has jobs' sizes smaller than 40 ut; the second rank is an intermediate range of sizes greater than 40 ut and smaller than or equal to 80 ut; and the last range has the jobs with service time larger than 80 ut. The graph at the right side of the figure is a pie-graph with the delay of the group of jobs served by each one of the three servers (mean service time 20 ut, 15 ut and 10 ut).

Jobs with residence times less than 2500 ut (Group 3) are represented by Fig. 58. The histogram at the left side of Fig. 58 shows the contribution of jobs' sizes to the system delay with a statistical mode clearly in the service time range of 15 ut. The graph with in the middle of Fig. 58 shows the jobs from 0 to 40 ut with the main contribution to the system

delay. The pie chart identifies jobs served by Server A (mean services time of 20 ut) and Server B (mean services time of 15 ut) with the most important contribution to the system delay.

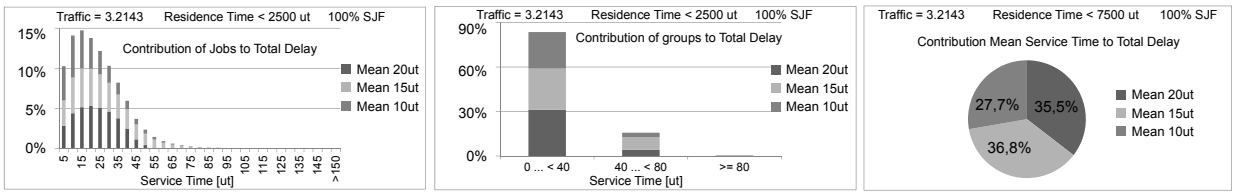


Figura 58 – The job's size identification into the group of TAVG < 2500 ut. The contribution to the system's delay through classes (at the left side), groups (the middle side) and mean service times (at the right side).

Jobs with residence times between 2500 ut and 7500 ut (within Group 2) are represented by Fig. 59.

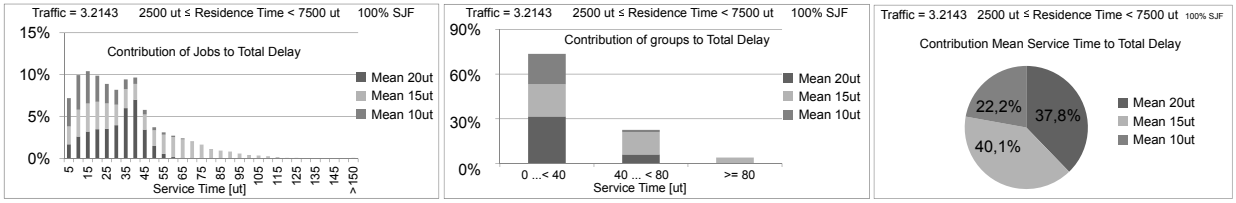


Figura 59 – The job's size identification into the group of 2500 ut ≤ TAVG < 7500 ut. The contribution to the system's delay through classes (at the left side), groups (the middle side) and mean service times (at the right side).

The histogram at the left of Fig. 59 shows the main statistical mode in the range of 15 ut and, a secondary mode in 40 ut. The graph in the middle of Fig. 59 shows jobs from 0 to 40 ut with the most important contribution to the system delay. The pie chart identifies the mean services of 20 ut (Server A) and 15 ut (Server B) are the most present in the Group 2. Jobs with residence times greater than 7500 ut (within Group 1) are represented by Fig. 60.

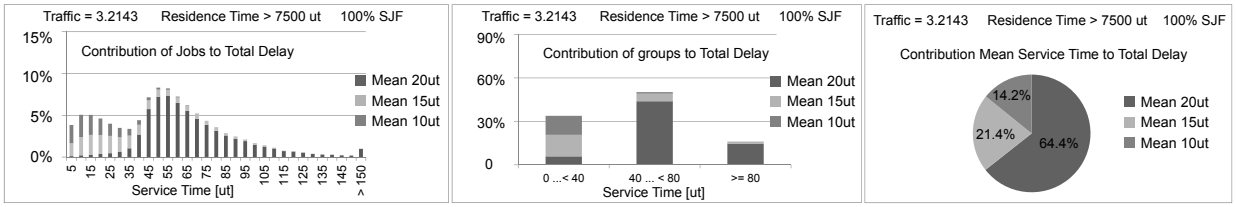


Figura 60 – The job's size identification into the group of TAVG ≥ 7500 ut. The contribution to the system's delay through classes (at the left side), groups (the middle side) and mean service times (at the right side).

The histogram at the left side of Fig. 60 shows the main statistical mode around the service time ranges 60 ut and 65 ut and, a secondary mode around the class 10-15 ut. The graph in the middle of Fig. 60 shows the jobs appears mostly from 40 ut to 80 ut range. The pie chart identifies the mean services of 20 ut (Server A) is the largest in Group 1. The graph at the right side of Figure 61 shows a long tail distribution for the total number of entities with the mode value around 60-70 ut.

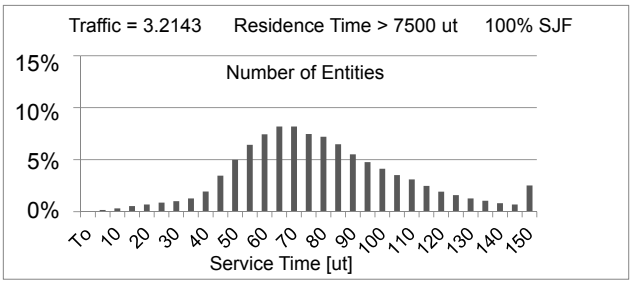


Figura 61 – The distribution for the total number of entities vs the percentage of delay into the group of Residence Time > 7500 ut.

The distribution at the right side of Figure 62 shows that from entities with total service times greater that 135 ut there are 7% of the total delay of entities within the range of residence time values greater than TAVG + 2*TSTD, and 6% from 0 to 45 ut.

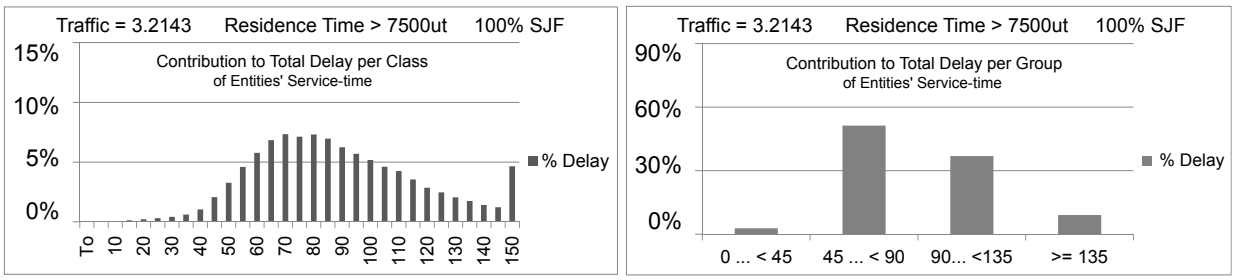


Figura 62 – The distribution of delays from entities' classes of total service times. At the left side, the histogram represents the general distribution of delay's contribution from each class of entities. The graph in the right side shows the entities' contribution to the total delay of the system from four groups of service time.

Note that the contribution of entities to the total delay (right side of Fig. 62) follows the same distribution of the number of entities (Fig.61). The histogram at the right side shows the entities' group with values 45-90 ut has the largest impact on the delay. These results are discussed in the following section.

6.7 Discussion

The proposed *Classifying model* (Section 6.3) uses the the conservation law of the total system delay from Kleinrock and the Little's law to associate the jobs'sizes to the system's delay. The model details the jobs' size distribution within the tail of the TAVG's distribution in Fig. 60, to show that more than 64% of the contribution to system's delay comes from the slowest server (Server A with mean 20 ut in the pie chart). It is relevant to note that the histogram at the left side of Fig. 60 shows service times mainly distributed around the mode within the interval from 45 to 55 ut (most jobs comes from Server A), but, also around a secondary mode with 36% of jobs around the 15 ut (most jobs come from Servers B and C). Such data do not consider the jobs' dependence from its entities, because of this is a multi-class system in which an entity has three jobs to be processed each one in a different server in random order. Those earlier jobs' information can not indicate if jobs from different servers belong to the same entity. Figures 61 and 62 complete these necessary information.

Fig. 62 compared to Fig. 61 shows the increased impact the largest sizes classes of entities on the system's delay. However, the service time classes greater than 55 ut in Figure 60 show a huge decrease in the delay's contribution of the largest jobs' sizes from the Server A (mean service time of 20 ut). This comparison of the Figures 62 and 60 drives us to the important impact from the Group 0-40 ut of the mean service-times 10ut (Server C) and 15 ut (Server B) have on the tail's delay (in addition to the contribution of the largest jobs from shortest means), as it can be seen through the graph in the middle of the Figure 60. It is clear, as already noted, that the distribution of the service time in this group is a characteristic long tail distribution (even with such a high traffic of 3.2143 E) with less than 4% of the service-times from TAVG + 2*TSTD (of the exponentially distributed service-times). Additionally, the distribution of the contribution of the jobs' sizes to the system delay is clearly correlated to the distribution of the job's sizes.

Through Figure 60 that represents the distribution's tail of the residence times with traffic of 3.2143 E, it is possible to verify that there is a need to perform effective management of the system. This should be carried out not only for jobs with extremely high values, but already for those with values above 40 ut. Note that 40 ut represents the $TAVG|_{B=20ut}$, for the slowest server, plus one standard deviation (20ut + 20ut), and 80ut (20+3.20) ut is $TAVG|_{B=20ut} + 3.TSTD$ − the standard deviation of B is equal to $TAVG$ because B is exponentially distributed. It is important to recall that the jobs are entity's dependent. Considering the main contribution to system's delay comes from 40ut $\leq B \leq$ 80 ut, they are mainly from mean service time of 20 ut, its is important to manage the largest jobs related to this server as shown in the pie chart of Fig 60. However, it is of great importance that the

largest jobs of the smallest mean service-times ($\geq$ *TAVG* mean) receive the same attention. The quantitative analysis by itself can drive a decision to be made, but the qualitative evaluation is of great importance to the best decision. Thus, this work reinforces the system monitoring/management application of this DES method as an improvement of the regular use of the queuing model for planning/sizing.

6.8 Conclusion

The classifying jobs method in two levels was shown. It identifies the jobs' contribution to the performance of a hybrid SJF/FIFO system submitted to high traffic in a JN configuration of parallel servers. The conservation of system's delay is used to associate the delay of the service-time's classes to the the respective classes of jobs' sizes. It was also shown that the distribution of the delay's contribution from jobs' sizes are correlated to the distribution of the jobs' sizes. This correlation also gives enough information to the importance of the shortest mean service time contribution to the highest TAVG's classes in the tail of the TAVG's distribution. Such conclusions are important to plan/size operations as a long-term perspective, and also it is of main importance for the managing operation as a short-term perspective. In the sense of the short-term perspective, it is an identifying model to plan and size the system performance around its estimated limits, implying in a method that can support managing actions. As a long-term perspective the model allows the identifying of classes of jobs that can the most reduces the performance (with the sense of the capacity of decreasing the system's flow of entities). There were no system's flow evaluation, but the method performed the identification of the classes of jobs the most related to the distribution of the delay's contribution for the system through the jobs's sizes in the time range from the TAVG's tail.

The next research step is to feed the SJF/FIFO's simulation model back with the results of the Classifying model to be compared to regular results. This work can be improved by a Fuzzy analyzer to evaluate some qualitative conditions, as e.g., consummation, fidelity and fairness perception. A suggestion for future work is the implementation of a parallel computing module to processes simultaneously the three activities of the classifier.

7 Results and Discussion

The results and contributions of each chapters are summarized in Table 11, In the next paragraphs, they are described and related to the central objectives of the thesis, i.e. planning through the mapping method, and managing through the control of the job's size.

Tabela 11 – Main contributions for each chapter

Chapter		Result	Contribution to the Thesis
Chap 2	2.1	Jackson network design	Model design
	2.2	Low-cost operation	Applicability
	2.3	Online operation	Applicability
	2.4	Management tool	Starting the central goal
Chap 3	3.1	Incremental Validation	Model design
	3.2	SJF Policy	Model design
	3.3	Variables trade-off	Threshold control
Chap 4	4.1	Instability and Predictability	Model design
	4.2	Hybrid Simulation DES/CSM	Model design
	4.3	Job Identification	Performance management
Chap 5	5.1	Hybrid Policy SJF/FIFO	Model complexity
	5.2	Kanban Policy Control	Model complexity
	5.3	Overloading Toggler	Performance management
	5.4	Mapping Method	Planning & Management
Chap 6	6.1	Double Classifying Distribution	Model design
	6.2	Planning/Dimensioning	Goal
	6.3	Online Management	Goal

Santos *et al.* (2018) contribute to this thesis with the validation of the implementation design through the Jackson Network Model (Table 11 - result.2.1). This framework is able to randomly and equiprobably assign either one job to N free servers or one free server to N queued jobs. This capability can be broadly applied to different industries as Services (e.g. health and call-center), Manufacturing (e.g. biscuits and packed food) and Logistics (for example, laden bagged commodities on trucks). It also adds value as an actual *management tool* (Table 11 - result.2.4) supported by a regular process that enters, seizes and leaves the DES model in a parallel configuration that can be easily implemented. This model can support a (Table 11 - result.2.2) a very general RFID/Internet communication infrastructure (e.g. sensors, tags and reader antennas), where the management level can update the evolving

data to easily monitoring all servers. Two technologies that may support this implementation are a *digital twin* (e.g., the DES model to synchronize the management and operation levels in real-time) and *RFIDs* (Table 11 - result.2.3) to capture the actual operation data and to feed the *digital twin mirror* application. Specifically, with the former, the collection of information system within physical operation feeds automation or human-machine interface to provide high value-added chain (Table 11 - result.2.3) that can increase the system performance. Notice that it is not necessary for all levels and/or servers to be close together because it is possible to remotely manage these different operations through a regular Internet channel (Table 11 - result.2.4). Santos *et al.* (2018) provide an example of *Humanitarian Support*, in the light of dangerous environments, as in wars and armed conflicts with personnel and infrastructure restrictions. This application could provide corrections and/or adjustments on either the resources availability or to diverting entities traffic to one specific server or any other strategic improvement. More recently, the support of IoT extended this perspective to meet different interface platforms (Apple iOS, Google Android, Windows Phone) and spatial location (Table 11 - result.2.3). This perspective serves to both planning and management objectives.

Moreover, Chapter 3, *Identifying the SJF and FIFO Compromise* contributes with the work on reaching better performance with the Jackson Network design (FIFO) by submitting it to the SJF policy (Table 11 - result.3.2). This design looks for improving the system flow in order to gain flexibility in the accommodation of changes, e.g., in the scheduling policies and instabilities of the traffic intensity. The need for knowing the jobs' sizes can be improved by the estimation of one task's length to choose the shortest job in a SJF policy system. The overcoming of response and performance limitations are examples of applications when the decision making is limited as the analytic models are out of the boundaries they were designed for. The SJF policy validated with the Jackson Network model is one example of this type of overcoming. This additional knowledge and gain allow unknown responses to become feasible and actually observed with the use of Incremental Validation − despite of the limits of the analytic models (Table 11 - result.3.1). This was shown with the example of the relationship between variables submitted to different policies (SJF and FIFO). SJF is negatively compromised with FIFO as shown in the relationship between TAVG and TMAX (trade-off), and with the positive compromise between TMAX and TSTD independent of the traffic intensity (Table 11 - result.3.3). One example of a key requirement is the ability to analyze the performance of a dynamic system where *the traffic intensity is variable* or unstable (Table 11 - result.4.1) as shown in the Chapter 4.

In the work *A New Framework for the Performance Analysis of the Single-Server Non-preemptive Scheduling under Varying Traffic Conditions*, it is shown that when the sys-

tem is submitted to different traffic intensities under the SJF policy, the DES/CSM hybrid simulation model is able to identify the size of a job that allows the system to satisfy a given QoS requirement (e.g. maximum mean time). This DES/CSM hybrid simulation model was implemented (Table 11 - result.4.2). The DES feeds the TAVG value to the CSM that, supported by an analytic model yelds the expected service time of the job (Table 11 - result.4.3). This implementation supports the application of a method based in the identification of jobs' sizes for improving systems performance.

The *Simulation Model for Performance Analysis of a Hybrid-Policy (SJF/FIFO) Parallel System* in Chapter 5 presents the dynamic toggling between the policies SJF and FIFO according to the number of jobs in the system (Table 11 - result.5.1). The toggling method addresses all the jobs by demand in a dynamic queue. This is a pull-production type model driven by a kanban-type logic (in the sense of the Just-in-time management) that changes the priority of the jobs according to the policy at the moment of a request (Table 11 - result.5.2). This clean and dynamic method randomly allocates a job to available servers or allocates one free server to a set of queued jobs despite the scheduling policy. The model is enhanced by toggling during overflow where the system can not remain in the regular inflow and outflow cycle of jobs. This improvement increases the system flow for a given time (Table 11 - result.5.3), thus stabilizing the number of entities below the system limit. This improvement facilitates the operations with shorter run times and approximates its actual performance to a theoretical threshold established by requirements as, e.g., the total system capacity or a kind of maximum time in the system. Therefore, this InV/DES method can estimate the system performance through different traffic intensities in a dataset. These data are mapped as a surface graph to allow both long-term planning and the management of operations in short-term. The long term mapping identifies (a priori) the limiting conditions to be avoided. The (online, if implemented) management functionality provides the previous adaptability to diverting the system (Table 11 - result.5.4) from bottlenecks.

The simulation model equipped with the two-level classifier module (named *Classifying*) allows the system to identify classes of jobs' sizes from any time-dependent variable. In the presented case study, the first level ranked the residence times. The second one arranged the service times within residence time intervals (Table 11 - result.6.1). The *Classifying* module evaluates (in a given interval) how much delay each service time range causes to the system (Table 11 - result.6.2). This capacity is relevant to reduce the impact of jobs on the system delay through either a human-machine interface or an automated solution (Table 11 - result.6.2). This capability improves the method on a three-dimensional identification of a chosen monitored variable (e.g., TMAX) to guide a decision maker through a map of estimated values from a surface graph of possible bottlenecks.

8 Conclusion

We presented a method of improving the performance of moderately complex queuing systems with a case study that manage the jobs' sizes of a SJF/FIFO hybrid-scheduling system submitted to instabilities, as for example increasing traffic conditions. The method identifies one typical range of jobs' sizes by the delay's contribution of this jobs' sizes to the total system delay. This range of Jobs' sizes is used to prevent the system from performing estimated bottleneck results.

A discrete-event simulation model that approaches the actual system through the use of incremental validation is designed to be implemented with any arrival or service probability distribution, and also with a large number of servers. This characteristic give a broad adaptability to the method that can map the estimated bottlenecks in a surface of responses for one required variable. The method is able to draw one graph to each one of a possible set of required variables. The planning graph can be used to compare the collected system responses (e.g., via RFID sensoring) to manage action (e.g., through human/machine or full-automated interfaces) according to the previous strategies of the system requirements.

This managing capability can be implemented either as a local operation, or a distant operation, as e.g., in other countries supported by RFID/WEB technologies (if they are available). The DES model is a regular enter-size-leave application of parallel servers that demand small data packages, and can be easily executed through an ordinary digital twin application (as suggested). Even multiple parallel server systems with a single discipline (FIFO) need management. For example, the analysts/managers need to evaluate the impact of the jobs' size and/or the maximum residence time (TMAX) on the system submitted to unexpected high traffic (~ 0.7 E per server). Therefore, to optimize the system it is important to trade-off variables; Depending on the stipulated goals and requirements, the system can improve performance with the hybrid SJF/FIFO discipline. It toggles SJF to FIFO to improve the FIFO's TAVG, and from FIFO back to SJF to decrease the SJF's TMAX.

The simpler queuing model with parallel servers introduces the planning and sizing capability. The Jackson queuing design added complexity and validation to the model as a first contribution, and the second is its actual application to the management facility. The model was capable of showing that this design aims at increasing the throughput, and also, that the throughput time is related to the slowest mean processing time of the servers.

This work presented the distribution of the SJF's residence times as a kind of adjustment (improvement) in the FIFO distribution. The shorter SJF's mean residence time

occurs together with a shorter half-width but with a kind of entities' loss that form the long tail. The complexity of the system evolves step-by-step through InV to reach the hybrid SJF/FIFO scheduling policy that extends the scope of the work. The improvement of the efficiency/effectiveness of the system took into account the average residence time (flow), the mean maximum average residence time (TMAX), and (in a first approach) the predictability of a given and estimated job's size.

The first proposal was the hybrid DES/CSM simulation framework. It showed an accurate and effective result for identifying the size of jobs that exceed a certain threshold. Recall that this framework confirmed a very small percentage of jobs with excessive duration (elephants). The same occurs to the ones with short duration (rats). That approach also made use of the concept of system management (not just planning and dimensioning). The final approach to improving the efficiency/effectiveness of the system took into account the impact of jobs on the system delay. This approach was necessary because the predictability is associated to pure SJF' policy and it could not be used as a threshold in hybrid-policy models. The new approach showed that the job's delay is positively correlated to the job's size, and so do the classes of jobs' sizes. This identification allowed the suggestion of this full model for either the planning/dimensioning, or managing the hybrid SJF/FIFO system. The results reported in Chapter 5 are expected to be improved by the selected jobs' size information as a feedback sent to managing the system.

Future Work

The application of this method to other real-world cases with actual (non-exponential) distributions may benefit from this initial work. In such cases, the management may be even more important to avoid extreme situations. . Work on the forecasting of the number of entities in the system (initially predicted, e.g. with a Kalman filter) through the traffic intensity is already underway. It may reach the point where the maximum average time reaches the established requirements. It is an important case to evaluate the next research step. The use of parallel processing where cores execute different activities of this method may be an interesting case to evaluate the iterative responses' behavior. Another possible research under evaluation is the application of Fuzzy logic to combine the use of qualitative variables for improving system performance through this hybrid SJF/FIFO model.

Bibliography

ADAN, I.; RESING, J. *Queueing Theory. [electronic resource]*. 1st. ed. Eindhoven, NB, Netherlands: Eindhoven University of Technology. Dep. of Math. and Comp. Sci., 2001, 2001. Disponível em: <https://books.google.com.br/books?id=dAViMwEACAAJ>. Citado na página 30.

ALAM, K.; SADDIK, A. E. C2ps: A digital twin architecture reference model for the cloud-based cyber-physical systems. *IEEE Access*, v. 5, p. 2050–2062, 2017. ISSN 21693536. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edselc&AN=edselc.2-52.0-85015767302&lang=pt-br&site=eds-live&scope=site>. Citado 2 vezes nas páginas 28 e 29.

ANDRESEN, K.; GRONAU, N. An approach to increase adaptability in erp systems. In: KHOSROW-POUR, M. (Ed.). *Proc. 16th Annual Information Resources Management Association International Conference, Apr,30 2005*. San Diego, CA, USA: Idea Group Inc. 2005, 2005. p. pp. 883–885. ISBN 1591408237. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsbl&AN=CN063069991&lang=pt-br&site=eds-live&scope=site>. Citado na página 42.

ARAÚJOJR., J. T. D. O enigma da política industrial no Brasil. *Brazilian Journal of Political Economy*, n. 3, p. 461, 2015. ISSN 1809-4538. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edssci&AN=edssci.S0101.31572015000300461&lang=pt-br&site=eds-live&scope=site>. Citado na página 21.

BANKS, J.; II, J. C.; NELSON, B. L.; NICOL, D. M. *Discrete-event System Simulation*. 3rd. ed. [S.l.]: Upper Saddle River, NJ : Person Education Inc., 2010., 2010. ISBN 9783540764540. Citado na página 28.

BANKS, J.; II, J. C.; NELSON, B. L.; NICOL, D. M. *Discrete-event System Simulation.* [S.l.]: Upper Saddle River, NJ : Person Education Inc., 2010., 2010. ISBN 9783540764540. Citado 2 vezes nas páginas 43 e 80.

BEEMSTERBOER, B.; LAND, M.; TEUNTER, R.; BOKHORST, J. Reprint of "integrating make-to-order and make-to-stock in job shop control". *International Journal of Production Economics*, v. 194, p. 3 – 12, 2017. ISSN 0925-5273. Special Issue: Innovations in Production Economics. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0925527317303481>. Citado na página 113.

BEKKER, R.; VIS, P.; DORSMAN, J.; MEI, R.; WINANDS, E. The impact of scheduling policies on the waiting-time distributions in polling systems. *Queueing Systems*, v. 79, n. 2, p. 145 – 172, 2015. ISSN 02570130. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=iih&AN=100353189&lang=pt-br&site=eds-live&scope=site>. Citado na página 45.

BITRAN, G. R.; MORABITO, R. Open queueing networks: Optimization and performance evaluation models for discrete manufacturing systems. *Production and Operations Management*, v. 5, n. 2, p. 46, 1996. ISSN 1059-1478. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsbl&AN=RN013995370&lang=pt-br&site=eds-live&scope=site>. Citado 4 vezes nas páginas 29, 30, 31 e 80.

BLINDER, A. S. Offshore: Big deal, or business as usual? p. 1 – 38, 01 2009. Originally presented at Alvin Hansen Symposium at Harvard University, May 2, 2007. Disponível em: <https://www.princeton.edu/~blinder/papers/07juneCEPSwp149.pdf>. Citado na página 21.

BOXMA, O.; ZWART, B. Tails in scheduling. *SIGMETRICS Perform. Eval. Rev.*, ACM, New York, NY, USA, v. 34, n. 4, p. 13–20, mar. 2007. ISSN 0163-5999. Disponível em: <http://doi-acm-org.ez88.periodicos.capes.gov.br/10.1145/1243401.1243406>. Citado na página 45.

BRANKE, J.; NGUYEN, S.; PICKARDT, C. W.; ZHANG, M. Automated design of production scheduling heuristics: A review. *IEEE Transactions on Evolutionary Computation*, v. 20, n. 1, p. 110–124, Feb 2016. ISSN 1089-778X. Citado 2 vezes nas páginas 84 e 105.

BUSINESSDICTIONARY.COM. *insourcing*. WebFinance Inc., 2018. Acessed in 05dec2018. Disponível em: <http://www.businessdictionary.com/definition/insourcing.html>. Citado na página 22.

BüKE, B.; CHEN, H. Stabilizing policies for probabilistic matching systems. *Queueing Systems*, v. 80, n. 1/2, p. 35 – 69, 2015. ISSN buke2015. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=iih&AN=102440064&lang=pt-br&site=eds-live&scope=site>. Citado na página 114.

CHUDZICKA, J. Insourcing as a New Trend in Global Business. *Foundations of Management*, v. 5, n. 2, p. 7 – 24, 2013. Disponível em: <https://content.sciendo.com/view/journals/fman/5/2/article-p7.xml>. Citado na página 22.

CORREA, V. H. C.; RAMOS, P. A precariedade do transporte rodoviário brasileiro para o escoamento da produção de soja do Centro-Oeste: situação e perspectivas. *Revista de Economia e Sociologia Rural, Vol 48, Iss 2, Pp 447-472 (2010)*, n. 2, 2010. ISSN 0103-2003. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsdoj&AN=edsdoj.01a53fa1b0ec4b30ae5d484cf400ef0b&lang=pt-br&site=eds-live&scope=site>. Citado na página 21.

COWDREY, K. W. G.; LANGE, J. de; MALEKIAN, R.; WANNEBURG, J.; JOSE, A. C. Applying Queueing Theory for the Optimization of a Banking Model. *JOURNAL OF INTERNET TECHNOLOGY*, 19, n. 2, p. 381–389, 2018. ISSN 1607-9264. Citado na página 45.

CRUZ, N.-N. de la; DADUNA, H. Optimal capacity allocation in a production–inventory system with base stock. *Annals of Operations Research*, p. 1–16, 2017. ISSN 15729338. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edselc&AN=edselc.2-52.0-85031906237&lang=pt-br&site=eds-live&scope=site>. Citado na página 28.

CRUZ, N.-N. de la; DADUNA, H. Optimal capacity allocation in a production–inventory system with base stock. *Annals of Operations Research*, p. 1–16, 2017. ISSN 15729338. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edselc&AN=edselc.2-52.0-85031906237&lang=pt-br&site=eds-live&scope=site>. Citado 2 vezes nas páginas 45 e 84.

DARLINGTON, S.; ANDREONI, M. Truckers' Strike Paralyzes Brazil Even as President Tries to Woo Investors. *The New York Times*, p. 7, 2018. ISSN 0362-4331. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsgao&AN=edsgcl.540639945&lang=pt-br&site=eds-live&scope=site>. Citado na página 21.

DEVORE, J. *Probability and Statistics for Engineering and the Sciences*. Thomson-Brooks/Cole, 2004. (International student edition). ISBN 0534466834. Disponível em: <https://books.google.lk/books?id=9mEwQQAACAAJ>. Citado 2 vezes nas páginas 43 e 80.

DEVORE, J.; BERK, K. *Modern Mathematical Statistics with Applications*. [S.l.]: Thomson Brooks/Cole, 2007. ISBN 9780534404734. Citado na página 44.

EECKHOUT, L. The internet of things revolution. *IEEE Micro*, v. 36, n. 6, p. 4–4, 2016. ISSN 0272-1732. Citado 2 vezes nas páginas 30 e 41.

EECKHOUT, L. The internet of things revolution. *IEEE Micro*, v. 36, n. 6, p. 4–4, Nov 2016. ISSN 0272-1732. Citado na página 83.

ELMOUGY, S.; SARHAN, S.; JOUNDY, M. A novel hybrid of shortest job first and round robin with dynamic variable quantum time task scheduling technique. *Journal of Cloud Computing*, Springer Verlag, v. 6, n. 1, 2017. ISSN 2192113X. Cited By 1. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85021051273&doi=10.1186>. Citado 2 vezes nas páginas 64 e 113.

ELSAFI, A. . . .; JAWAWI, D. . . .; ABDELMABOUD, A. . . .; ALI, A. . . . A comparative evaluation of state-of-the-art integration testing techniques of component-based software. *Journal of Theoretical and Applied Information Technology*, v. 71, n. 2, p. 257–267, 2015. ISSN 18173195. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edselc&AN=edselc.2-52.0-84921316635&lang=pt-br&site=eds-live&scope=site>. Citado na página 44.

ENPNEWSWIRE, J. Fitch: Brazil Truck Strike May Drive Higher Bank Loan Losses. *ENP Newswire*, 2018. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsgao&AN=edsgcl.541291224&lang=pt-br&site=eds-live&scope=site>. Citado na página 21.

ENPNEWSWIRE, J. Truckers' Strike In Brazil Cripples The Country. *Morning Edition*, 2018. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsgao&AN=edsgcl.540629106&lang=pt-br&site=eds-live&scope=site>. Citado na página 21.

FEKI, M. A.; KAWSAR, F.; BOUSSARD, M.; TRAPPENIERS, L. The internet of things: The next technological revolution. *Computer*, v. 46, n. 2, p. 24–25, 2013. ISSN 0018-9162. Citado na página 41.

FEKI, M. A.; KAWSAR, F.; BOUSSARD, M.; TRAPPENIERS, L. The internet of things: The next technological revolution. *Computer*, v. 46, n. 2, p. 24–25, Feb 2013. ISSN 0018-9162. Citado na página 83.

FERNANDES, S. d. C.; SILVA, L.; PAIXãO, J.; ROSALEM, V. Insourcing as a Business Strategy. *Enciclopedia Biosfera*, v. 13, n. 24, p. 1809, 2016. Disponível em: <http://www.conhecer.org.br/enciclop/2016b/sociais/desterceirizacao.pdf>. Citado na página 22.

GOYAL, N.; FUSSELL, S. R. Intelligent interruption management using electro dermal activity based physiological sensor for collaborative sensemaking. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, ACM, v. 1, n. 3, p. 52:1–52:21, 2017. ISSN 2474-9567. Disponível em: <http://doi.acm.org/10.1145/3130917>. Citado na página 40.

GROSOF, I.; SCULLY, Z.; HARCHOL-BALTER, M. Srpt for multiserver systems. *Performance Evaluation*, v. 127-128, p. 154 – 175, 2018. ISSN 0166-5316. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0166531618302773>. Citado na página 45.

GUDA, H.; DAWANDE, M.; JANAKIRAMAN, G.; JUNG, K. S. Optimal policy for a stochastic scheduling problem with applications to surgical scheduling. *Production and Operations Management*, v. 25, n. 7, p. 1194, 2016. ISSN 1059-1478. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsgao&AN=edsgcl.457246960&lang=pt-br&site=eds-live&scope=site>. Citado na página 64.

GUNTHER, O. P.; KLETTI, W.; KUBACH, U. *RFID in Manufacturing. [electronic resource]*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2008., 2008. ISBN 9783540764540. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=cat05684a&AN=spr.9783540764540&lang=pt-br&site=eds-live&scope=site>. Citado na página 29.

HÄHNLE, R.; MUSCHEVICI, R. Towards incremental validation of railway systems. In: MARGARIA, T.; STEFFEN, B. (Ed.). *Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications*. Cham: Springer International Publishing, 2016. p. 433–446. ISBN 978-3-319-47169-3. Citado 4 vezes nas páginas 43, 44, 48 e 94.

HARRELL, C.; BOWDEN, R.; GHOSH, B. K. *Simulation Using Promodel*. 3rd. ed. [S.l.]: McGraw-Hill Higher Education, 2012. ISBN 0073401300. Citado na página 80.

HAYNES, S. N.; LENCH, H. C. Incremental validity of new clinical assessment measures. *Psychological Assessment*, American Psychological Association, Haynes, Stephen N.: Dept of Psychology, U Hawaii Manoa, 2430 Campus Road, Honolulu, HI, US, 96822, sneil@hawaii.edu, v. 15, p. 456 – 466, 2003. ISSN 1939-134X(Electronic),1040-3590(Print). Citado 3 vezes nas páginas 43, 48 e 94.

HUM, S.-H.; PARLAR, M.; ZHOU, Y. Measurement and optimization of responsiveness in supply chain networks with queueing structures. *European Journal of Operational Research*, v. 264, n. 1, p. 106 – 118, 2018. ISSN 0377-2217. Disponível em: <http://www.sciencedirect.com/science/article/pii/S037722171730437X>. Citado 2 vezes nas páginas 45 e 84.

IBRAHIM, L.; BRADFORD, B.; COLE, D.; LABRUYERE, L.; LEINNEWEBER, H.; PISZCZEK, D.; REED, N.; RYMOND, M.; SMITH, D.; VIRGA, M.; WELLS, C. *FAA-iCMM Version 2.0: An Integrated Capability Maturity Model for Enterprise-wide Improvement*. 2001. U.S. FAA Federal Aviation Administration, Sep 2001. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.625.5081&rep=rep1&type=pdf>. Citado na página 43.

J, S. R.; MAHMOUD, S. F. I.; EID, D. S.; J, S. N.; ALI, S. R.; HANNAH, D. Benchmarking of tqm practices in ingos: a literature review. *Benchmarking: An International Journal*, n. 1, p. 236, 2016. ISSN 1463-5771. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsemr&AN=edsemr.10.1108.BIJ.02.2015.0013&lang=pt-br&site=eds-live&scope=site>. Citado na página 27.

JACKSON, J. R. Networks of waiting lines. *Operations Research*, n. 4, p. 518, 1957. ISSN 0030364X. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsjsr&AN=edsjsr.167249&lang=pt-br&site=eds-live&scope=site>. Citado 2 vezes nas páginas 28 e 33.

JANIĆ, M. *Advanced Transport Systems: Analysis, Modeling, and Evaluation of Performances*. 1. ed. [S.l.]: Springer-Verlag London, 2014, 408 pages, 2014. v. 2014. ISBN 978-1-4471-6286-5,978-1-4471-6287-2. Citado 2 vezes nas páginas 50 e 95.

KIM, S.; KIM, S. Differentiated waiting time management according to patient class in an emergency care center using an open jackson network integrated with pooling and prioritizing. *Annals of Operations Research*, v. 230, n. 1, p. 35 – 55, 2015. ISSN 02545330. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=iih&AN=103107171&lang=pt-br&site=eds-live&scope=site>. Citado na página 28.

KIM, S.; KIM, S. Differentiated waiting time management according to patient class in an emergency care center using an open jackson network integrated with pooling and prioritizing. *Annals of Operations Research*, v. 230, n. 1, p. 35 – 55, 2015. ISSN 02545330. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=iih&AN=103107171&lang=pt-br&site=eds-live&scope=site>. Citado 2 vezes nas páginas 45 e 83.

KLEINROCK, L. A conservation law for a wide class of queueing disciplines. *Naval Research Logistics Quarterly*, v. 12, n. 2, p. 181–192, 1965. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800120206>. Citado 2 vezes nas páginas 53 e 116.

KUA, J.; NGUYEN, S. H.; ARMITAGE, G.; BRANCH, P. Using active queue management to assist iot application flows in home broadband networks. *IEEE Internet of Things Journal*, v. 4, n. 5, p. 1399–1407, 2017. Citado na página 40.

LAMOLLE, M.; MENET, L.; DUC, C. L. Incremental checking of master data management model based on contextual graphs. *Enterprise Information Systems*, v. 9, n. 7, p. 681–708, 2015. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84929130229&doi=10.1080%2f17517575.2013.792395&partnerID=40&md5=1a6b0ff6da9b4ae5aebed0366b03fa32>. Citado na página 44.

LEADERSHIP, B. *From outsourcing to insourcing in another country*. 2016. Acessed in dec04,2018. Disponível em: <http://www.leadership-bc.com/en/index.php/estrategia-e-financas/item/105-de-outsourcing-para-insourcing-noutro-pais.html>. Citado na página 22.

LEITE, P. R. E.; MARTINS, P. S.; URSINI, E. L. A proposal for performance analysis and dimensioning of iot networks. In: *BTSym 2017 - Brazilian Technology Symposium (Dec 2017)*. Campinas - São Paulo, Brazil: [s.n.], 2017. ISSN 2447-8326. Disponível em: <http://lcv.fee.unicamp.br/images/BTSym-17/Papers/76916.pdf>. Citado 2 vezes nas páginas 45 e 47.

LEYE, S.; EWALD, R.; UHRMACHER, A. M. Composing problem solvers for simulation experimentation: A case study on steady state estimation. *PLOS ONE*, Public Library of Science, v. 9, n. 4, p. 1–13, 04 2014. Disponível em: <https://doi.org/10.1371/journal.pone.0091948>. Citado 2 vezes nas páginas 59 e 94.

LITTLE, J. D. C. A proof for the queuing formula: L = lambda/w. *Oper. Res.*, INFORMS, v. 9, n. 3, p. 383–387, jun. 1961. ISSN 0030-364X. Disponível em: <http://dx.doi.org/10.1287/opre.9.3.383>. Citado na página 116.

LOLLI, F.; GAMBERINI, R.; GIBERTI, C.; RIMINI, B.; BONDI, F. A simulative approach for evaluating alternative feeding scenarios in a kanban system. *International Journal of Production Research*, v. 54, n. 14, p. 4228 – 4239, 2016. ISSN 00207543. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=fst&AN=118193655&lang=pt-br&site=eds-live&scope=site>. Citado na página 93.

MARTINS, A. J. *RE-INSOURCING DA LOGÍSTICA: ESTUDO DE CASOS MÚLTIPLOS EM CENTROS DE DISTRIBUIÇÃO DE PEÇAS DE REPOSIÇÃO*. Dissertação (Mestrado) — Faculdade de Engenharia, Arquitetura e Urbanismo, UNIMEP, Santa Bárbara d'Oeste, SP, Brasil, 2016. Mestrado em Engenharia de Produção. Citado na página 22.

MARTINS, P. S.; BURNS, A. On the meaning of modes in uniprocessor real-time systems. In: *Proceedings of the 2008 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2008. (SAC '08), p. 324–325. ISBN 978-1-59593-753-7. Disponível em: <http://doi.acm.org/10.1145/1363686.1363770>. Citado 2 vezes nas páginas 62 e 80.

MLINAR, T.; CHEVALIER, P. Pooling heterogeneous products for manufacturing environments. *4OR*, v. 14, n. 2, p. 173–200, Jun 2016. ISSN 1614-2411. Disponível em: <https://doi.org/10.1007/s10288-016-0307-1>. Citado 2 vezes nas páginas 13 e 95.

MOR, H. Queueing disciplines. In: _____. *Wiley Encyclopedia of Operations Research and Management Science*. American Cancer Society, 2010. p. 1–11. ISBN 9780470400531. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470400531.eorms0699>. Citado 5 vezes nas páginas 24, 65, 81, 84 e 112.

NOAM, K.; GELBARD, R. Knowledge Sharing Motivation Among External and Internal IT Workers. *Journal of Information & Knowledge Management*, v. 17, p. 1850026, 08 2018. Citado na página 22.

OGLOBO. *Acao social na Vila Kennedy realizou mais de 13 mil atendimentos*. 2018. Online. Disponível em: <https://oglobo.globo.com/rio/acao-social-na-vila-kennedy-realizou-mais-de-13-mil-atendimentos-22504952>. Acesso em: 27 nov. 2018. Citado 3 vezes nas páginas 21, 27 e 32.

OGLOBO. Greve dos caminhoneiros pode afetar trajetória dos juros. 2018. Disponível em: <http://link.galegroup.com/apps/doc/A544397462/AONE?u=capes&sid=AONE&xid=0a95bf77>. Acesso em: 27 nov. 2018. Citado na página 21.

OHNO, T. *Taiichi Ohno's Workplace Management*. Mukilteo, WA: Genba, c2009., 2009. ((Business/Management)). ISBN 9780978638757. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=cat04198a&AN=unicamp.000478929&lang=pt-br&site=eds-live&scope=site>. Citado 2 vezes nas páginas 55 e 93.

OWAIDAT, B.; NASSAR, H.; KASSEM, R. A shortest job first (sjf)-like scheme for efficient call handoff in mobile networks. In: *2015 Fifth International Conference on Digital Information and Communication Technology and its Applications (DICTAP)*. [S.l.: s.n.], 2015. p. 216–221. Citado na página 64.

OXFORDDICTIONARIES.COM. *insourcing*. The Oxford University Press, 2018. Acessed in 05dec2018. Disponível em: <https://en.oxforddictionaries.com/definition/insourcing>. Citado na página 22.

PEDRO, P. *Schedulability of mode changes in flexible real-time distributed systems*. Tese (Doutorado) — The University of York, Deramore Lane, University of York, Heslington, York, YO10 5GH, UK, nov. 1999. York computer science thesis; 99/11. Citado na página 62.

PEDRO, P. S.; BURNS, A. Schedulability analysis of mode changes in flexible real-time systems. In: *Proceedings of the 10th Euromicro Conference on Real Time Systems*. [S.l.: s.n.], 1998. Citado na página 62.

PERA, T.; ROCHA, F.; NETO, S. da S.; CAIXETA-FILHO, J. *Análise dos impactos da Medida Provisória nº 832 de 2018 (Política de Preços Mínimos do Transporte Rodoviário de Cargas) na logística do agronegócio brasileiro*. 2018. Citado 3 vezes nas páginas 15, 22 e 23.

PIPLANI, R.; ANG, A. W. H. Performance comparison of multiple product kanban control systems. *International Journal of Production Research*, Taylor & Francis, v. 56, n. 3, p. 1299–1312, 2018. Disponível em: <https://doi.org/10.1080/00207543.2017.1332436>. Citado na página 93.

QUINTARELLI, E.; RABOSIO, E.; TANCA, L. A principled approach to context schema evolution in a data management perspective. *Information Systems*, v. 49, p. 65 – 101, 2015. ISSN 0306-4379. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0306437914001847>. Citado 2 vezes nas páginas 44 e 47.

REIS, E.; FERNANDES, S.; BORETO, M.; ROSALEM, V.; SANTOS, V. An Insourcing Practice in Organizations: Integration Review. *Enciclopedia Biosfera*, v. 15, n. 28, p. 1421/1433, 2018. Disponível em: <http://www.conhecer.org.br/enciclop/2018B/SOC/apratica.pdf>. Citado na página 22.

REKHA, R. Thinking about Insourcing? *Journal of Information & Knowledge Management*, PWC India 2017, 2017. Disponível em: <https://www.pwc.in/consulting/management-consulting/shared-services-outsourcing-advisory/perspectives/thinking-about-insourcing.html>. Citado na página 22.

RICHARDSON, D. A.; LEEUW, S.; DULLAERT, W. Factors affecting global inventory prepositioning locations in humanitarian operations-a delphi study. *Journal of Business Logistics*, v. 37, n. 1, p. 59 – 74, 2016. ISSN 07353766. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=bsx&AN=113902210&lang=pt-br&site=eds-live&scope=site>. Citado na página 27.

SANDMANN, W. Benefits of alternating fcfs/sjf service order. In: *Proceedings of the 6th WSEAS International Conference on Applied Informatics and Communications*. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2006. (AIC'06), p. 194–199. ISBN 960-8457-51-3. Disponível em: <http://dl.acm.org/citation.cfm?id=1366421.1366454>. Citado 3 vezes nas páginas 45, 83 e 106.

SANDMANN, W. Quantitative fairness for assessing perceived service quality in queues. *Operational Research*, v. 13, n. 2, p. 153–186, Jul 2013. ISSN 1866-1505. Disponível em: <https://doi.org/10.1007/s12351-011-0111-9>. Citado 4 vezes nas páginas 52, 64, 105 e 114.

SANTOS, G.; TEIXEIRA, A. Insourcing as a Strategy for Cost Reduction: a Methodology to Identifiy and Mensurate the Supporting Decition Factors. *Revista Sociedade, Contabilidade e Gestão*, Universidade Federal do Rio de Janeiro, v. 10, n. 2, p. 36/53, 2015. ISSN 1982-7342. Disponível em: <http://www.atena.org.br/revista/ojs-2.2.3-06/index.php/ufrj/article/viewFile/2551/2161>. Citado na página 22.

SANTOS, V.; URSINI, E.; MARTINS, P.; YACOUB, M. A management tool based on discrete event simulation for humanitarian support. In: RABE, M.; JUAN, A.; MUSTAFEE, N.; SKOOG, A.; JAIN S., J. B. (Ed.). *Proceedings of the 2018 Winter Simulation Conference*. Gothenburg, Sweden, Sweden: IEEE, 2018. p. 45–56. ISSN 1558-4305. Disponível em: <https://ieeexplore-ieee-org.ez88.periodicos.capes.gov.br/document/8632484>. Citado 13 vezes nas páginas 10, 12, 22, 46, 47, 51, 62, 81, 83, 84, 112, 123 e 124.

SANTOS, V. F. *"Principles of the method of dynamic sizing of operations with real time queues". [in portuguese]*. Dissertação (Mestrado) — School of Technology, University of Campinas, Limeira, SP, Brazil, 2014, 2014. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=cat04198a&AN=unicamp.000927509&lang=pt-br&site=eds-live&scope=site>. Citado 3 vezes nas páginas 42, 83 e 84.

SCHLUSE, M.; PRIGGEMEYER, M.; ATORF, L.; ROSSMANN, J. Experimentable digital twins 8212;streamlining simulation-based systems engineering for industry 4.0. *IEEE Transactions on Industrial Informatics*, v. 14, n. 4, p. 1722–1731, April 2018. ISSN 1551-3203. Citado 2 vezes nas páginas 28 e 40.

SCHLUSE, M.; PRIGGEMEYER, M.; ATORF, L.; ROSSMANN, J. Experimentable digital twins 8212;streamlining simulation-based systems engineering for industry 4.0.

IEEE Transactions on Industrial Informatics, v. 14, n. 4, p. 1722–1731, April 2018. ISSN 1551-3203. Citado 2 vezes nas páginas 50 e 93.

SERWADDA, A.; PHOHA, V. V. When mice devour the elephants: A ddos attack against size-based scheduling schemes in the internet. *Computers & Security*, v. 53, p. 31 – 43, 2015. ISSN 0167-4048. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0167404815000644&lang=pt-br&site=eds-live&scope=site>. Citado na página 112.

SPEAR, M. F.; MARATHE, V. J.; SCHERER, W. N.; SCOTT, M. L. Conflict detection and validation strategies for software transactional memory. In: DOLEV, S. (Ed.). *Distributed Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 179–193. ISBN 978-3-540-44627-9. Citado na página 44.

SUN, G.; WANG, G.; LIU, G. Air-interface slice based dynamic resource reservation for ultra-low-latency iot transmissions. In: *2016 IEEE 41st Conference on Local Computer Networks (LCN)*. [S.l.: s.n.], 2016. p. 603–606. Citado na página 112.

TANG, W.; REN, D.; LAN, Z.; DESAI, N. Toward balanced and sustainable job scheduling for production supercomputers. *Parallel Computing*, v. 39, n. 12, p. 753 – 768, 2013. ISSN 0167-8191. Programming models, systems software and tools for High-End Computing. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167819113000999>. Citado na página 63.

URSINI, E.; MARTINS, P.; TIMOTEO, V.; MASSARO, F. Modeling and simulation applied to link dimensioning of stream ip traffic with incremental validation. In: YILMAZ, L.; CHAN, W. K. V.; MOON, I.; ROEDE, T. M. K.; MACAL, C.; ROSSETTI, M. D. (Ed.). *Proc. 2015 Winter Simulation Conference, Dec 2015*. Huntington Beach, California: IEEE, 2016. p. pp. 3049–3060. ISBN 978-1-4673-9743-8/15. Citado 2 vezes nas páginas 44 e 47.

USDA, U. S. D. O. A. *Grain Transportation Report 2017*. The U.S. DEPARTMENT OF AGRICULTURE, Agriculture Marketing Services, 2017. 22 p. Acessed in 05dec2018. Disponível em: <http://dx.doi.org/10.9752/TS056.11-30-2017>. Citado na página 23.

USDA, U. S. D. O. A. *Grain Transportation Report 2018*. The U.S. DEPARTMENT OF AGRICULTURE, Agriculture Marketing Services, 2018. 22 p. Acessed in 05dec2018. Disponível em: <http://dx.doi.org/10.9752/TS056.11-29-2018>. Citado na página 23.

VENKATAPATHY, A. K. R.; BAYHAN, H.; ZEIDLER, F.; HOMPEL, M. ten. Human machine synergies in intra-logistics: Creating a hybrid network for research and technologies. In: *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*. Prague, Czech Republic, 3 - 6 September, 2017: [s.n.], 2017. p. 1065–1068. Citado 2 vezes nas páginas 28 e 29.

VENKATAPATHY, A. K. R.; BAYHAN, H.; ZEIDLER, F.; HOMPEL, M. ten. Human machine synergies in intra-logistics: Creating a hybrid network for research and technologies. In: *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*. [S.l.: s.n.], 2017. p. 1065–1068. Citado na página 84.

WARNER, M.; HEFETZ, A. Insourcing and Outsourcing The Dynamics of Privatization Among U.S. Municipalities 2002-2007. *JOURNAL- AMERICAN PLANNING ASSOCIATION*, n. 3, p. 313, 2012. ISSN 0194-4363. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=edsbl&AN=RN318441039&lang=pt-br&site=eds-live&scope=site>. Citado na página 22.

WHITE, L. J.; LEUNG, H. K. N. Integration testing. *Encyclopedia of Software Engineering*, v. 1, n. 2nd edition, p. 638 – 641, 2002. ISSN 9780471210085. John Wiley and Sons Inc. Jan.2002. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=iih&AN=19487151&lang=pt-br&site=eds-live&scope=site>. Citado na página 44.

WIERMAN, A.; HARCHOL, M. B. Classifying scheduling policies with respect to higher moments of conditional response time. *ACM SIGMETRICS Performance Evaluation Review*, ACM, v. 33, n. 1, p. 229–240, 2005. Citado 5 vezes nas páginas 24, 64, 66, 75 e 76.

WIERMAN, A.; WINANDS, E. M.; BOXMA, O. J. Scheduling in polling systems. *Performance Evaluation*, Elsevier, v. 64, n. 9-12, p. 1009–1028, 2007. Citado 2 vezes nas páginas 64 e 113.

WU, X.; XIE, L. On load balancing strategies for baggage screening at airports. *Journal of Air Transport Management*, v. 62, p. 82 – 89, 2017. ISSN 0969-6997. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0969699716303313>. Citado na página 114.

XOXA, N.; ZOTAJ, M.; TAFA, I.; FEJZAJ, J. Simulation of first come first served (fcfs) and shortest job first (sjf) algorithms. *IJCSN-International J. Comput. Sci. Netw. ISSN*, v. 3, n. 6, p. 2277–5420, 2014. Citado na página 63.

XU, B.; XU, L. D.; CAI, H.; XIE, C.; HU, J.; BU, F. Ubiquitous data accessing method in iot-based information system for emergency medical services. *IEEE Transactions on Industrial Informatics*, v. 10, n. 2, p. 1578–1586, 2014. ISSN 1551-3203. Citado na página 29.

XU, B.; XU, L. D.; CAI, H.; XIE, C.; HU, J.; BU, F. Ubiquitous data accessing method in iot-based information system for emergency medical services. *IEEE Transactions on Industrial Informatics*, v. 10, n. 2, p. 1578–1586, May 2014. ISSN 1551-3203. Citado na página 83.

XU, X. From cloud computing to cloud manufacturing. *Robotics and Computer-Integrated Manufacturing*, v. 28, n. 1, p. 75 – 86, 2012. ISSN 0736-5845. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0736584511000949>. Citado 2 vezes nas páginas 50 e 93.

YEE, J. T.; OH, S.-C. *Technology Integration to Business. [electronic resource] : Focusing on RFID, Interoperability, and Sustainability for Manufacturing, Logistics, and Supply Chain Management.* London : Springer London : Imprint: Springer, 2013., 2013. ISBN 9781447143901. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true&db=cat04198a&AN=unicamp.001049029&lang=pt-br&site=eds-live&scope=site>. Citado na página 21.

ANEXO A – Full Paper Presented during The Winter Simulation Conference 2018.

Please refers to URL: <https://ieeexplore.ieee.org/stamp/stam.jsp?tp=&arnumber=8632484&isnumber=8632166>

ANEXO B – Full Paper Submitted to The Journal of Scheduling (JOSH) Springer/Nature

Journal of Scheduling

A New Framework for the Performance Analysis of the Single-Server Non-preemptive Scheduling under Varying Traffic Conditions

--Manuscript Draft--

Manuscript Number:	JOSH-D-18-00196
Full Title:	A New Framework for the Performance Analysis of the Single-Server Non-preemptive Scheduling under Varying Traffic Conditions
Article Type:	Paper
Keywords:	JAMT-D-18-04748 modes of operations; mode changes; Shortest Jobe First; scheduling; real-time management
Corresponding Author:	Vlademir Fazio Santos, MTech Universidade Estadual de Campinas Campinas, SP, BRAZIL
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Universidade Estadual de Campinas
Corresponding Author's Secondary Institution:	
First Author:	Vlademir Fazio Santos, MTech
First Author Secondary Information:	
Order of Authors:	Vlademir Fazio Santos, MTech
	Edson Luiz Ursini, MS 5.3 - RDIDP Livre Docente
	Paulo Sergio Martins Pedro, MS 3
	Michel Daoud Yacoub, Full Professor
Order of Authors Secondary Information:	
Funding Information:	